TREBALL FINAL DE GRAU

TÍTOL: Implementació d'un servidor per a l'anàlisi i visualització de l'estat de la xarxa de l'EPSEVG

AUTOR: Marc Ramiro Ramos

TITULACIÓ: Grau en Enginyeria Informàtica

DIRECTOR: Daniel Guasch Murillo

DEPARTAMENT: Enginyeria Telemàtica

DATA: 25 de Maig de 2015

TÍTOL: Implementació d'un servidor per a l'anàlisi i visualització de l'estat de la xarxa de l'EPSEVG

COGNOMS: Ramiro Ramos **NOM:** Marc

TITULACIÓ: Grau en Enginyeria Informàtica

PLA: 2010

DIRECTOR: Daniel Guasch Murillo

DEPARTAMENT: Enginyeria Telemàtica

QUALIFICACIÓ DEL TFG

TRIBUNAL

PRESIDENT	SECRETARI	VOCAL
Rafael Morillas	Xavier Masip	Jose Antonio Roman

DATA DE LECTURA: 25 de Maig de 2015

Aquest projecte té en compte aspectes mediambientals: □ Sí X No

TREBALL FINAL DE GRAU

RESUM (màxim 50 línies)

En este proyecto se pretende crear una herramienta de monitorización de redes de área local para su uso en la EPSEVG. Esta, debe ser capaz de generar estadísticas de uso de los equipos detectados, recibir alertas vía email y visualizar en cualquier momento, los equipos que se encuentran conectados a las redes monitorizadas. Para lograr estos objetivos, se ha desarrollado una aplicación web que permite crear sistemas de escaneado personalizables. Gracias a este sistema, se consigue monitorizar la red de aulas informáticas de la escuela en los distintos horarios establecidos.
Dentro de la aplicación, se ha implementado un sistema de clasificación de equipos, permitiendo agrupar los distintos dispositivos detectados en grupos para su mejor gestión. Con este sistema de clasificación, se ha podido proceder a la creación de un sistema de generación de estadísticas por grupos, permitiendo analizar el uso que se les da a las máquinas de las distintas aulas informáticas de la escuela, analizando previamente el uso individual de cada una de ellas.
Con la información recopilada por los distintos sistemas de escaneado, se ha desarrollado un sistema de alertas que permite a los administradores recibir notificaciones por correo.
Para que los administradores puedan interactuar con la aplicación, se han desarrollado dos interfaces distintas. La interfaz por defecto cuenta con un diseño minimalista y amigable que permite localizar de forma rápida los sistemas de escaneado configurados y todos los equipos detectados por ellos. La interfaz de administración cuenta con acceso a todas las configuraciones y datos recopilados por la aplicación. Desde esta última, los administradores pueden exprimir al máximo la aplicación.

Para que todo lo anterior haya sido posible, la aplicación web se ha integrado en un servidor dedicado dentro de la EPSEVG. Durante el proyecto se han analizado que componentes se requerían para que la aplicación pudiese ser funcional. Cuándo estos han sido seleccionados, se ha procedido a la integración de los mismos en el servidor, configurándolos para que trabajasen con la aplicación desarrollada.

Paraules clau (màxim 10):

GNU/Linux	PostgreSQL	Python 3	Django
Nmap	Nginx	Monitorización	Estadísticas
Alertas	Redes		

FINAL GRADE PROJECT

ABSTRACT (50 lines maximum)

This project aims to create a monitoring tool for LANs use at EPSEVG. This must be able to estimate usage statistics of all the discovered computers, and to send alerts via email. It also must provide an interface that allows to monitor all the devices connected on the monitored network. To achieve these goals, a web application that lets you to create customizable scanning systems has been developed. With this system, it is possible to monitor all the systems on the school at various times established.

Within the application, a classification system has been implemented allowing to group different devices in a customized groups, for its better management. With this classification system, it is possible to proceed with the creation of usage statistics by groups, previously analyzed by its devices.

With the information gathered by different scanning systems, it has developed an alert system that allows administrators to receive notifications by mail.

So administrators can interact with the application, two different web interfaces have been developed. The default interface has a minimalist and friendly design that allows quickly locate active devices. The administration interface has access to all settings and data collected by the application.

For all this has been possible, the web application is built on a dedicated server at EPSEVG. During the project development, several tools have been analyzed so which of them would made the application functional. When all of the tools have been selected, I proceeded to the integration in the server, setting them to work with the developed application

Keywords (10 maximum):

GNU/Linux	PostgreSQL	Python 3	Django
Nmap	Nginx	Monitoring	Statistics
Alerts	LAN		

Agradecimientos

Me gustaría agradecer a mi tutor, Daniel Guasch y a Jordi Enric de los servicio TIC de la EPSEVG, por confiar en mi para el desarrollo de este proyecto.

Agradezco a mi familia y a mi pareja, todo el apoyo y la ayuda que me han proporcionado a lo largo de este proyecto y a lo largo de toda mi carrera, sin ellos no habría sido posible llegar hasta aquí.

Por último, quiero agradecer la ayuda y el soporte que me ha ofrecido Fran Martín durante este tiempo.

ÍNDICE

1 INTRODUCCIÓN

Actualmente existe una gran cantidad de dispositivos distintos, capaces de acceder a las diferentes redes. Esto provoca que la administración de las redes, hoy en día, no sea para nada trivial. El dinamismo en la red producido por toda esta serie de dispositivos, hace que los administradores de redes, deban contar con diversas herramientas capaces de facilitarles la gestión, ayudándoles a mantenerlas bajo control y previniéndolas de los posibles peligros que pudiesen surgir. Además algunas de estas herramientas les permiten visualizar el estado de todos los sistemas de la red en cualquier momento.

Una de las tareas que llevan a cabo los administradores de redes, se trata del rastreo continuo de los dispositivos que se encuentran en ellas. A esta tarea se la conoce como monitorización. Mediante un procesado de los datos analizados en la red, se puede extraer cierta información que proporciona conocimiento, sobre los sistemas que se encuentran en ese momento conectados. Además de conocer que sistemas se encuentran presentes, se puede obtener información detallada de cada uno de ellos. Por ejemplo, es posible conocer que servicios se están ejecutando en un equipo e incluso que sistema operativo se está usando. Según la información recopilada de los dispositivos, los administradores pueden tomar decisiones consecuentes al uso.

El propósito de este trabajo de fin de grado, es el de desarrollar una de estas herramientas de monitorización y su despliegue en un servidor dedicado, con la configuración adecuada. Esta herramienta, estará dotada de varias funcionalidades que permitirán a los administradores de las redes, conocer el uso que se les da a los equipos monitorizados e incluso recibir alertas, informando que sistemas se encuentran activos en un rango determinado de horas.

2 MOTIVACIÓN

La motivación que ha llevado al desarrollo de este trabajo de fin de grado, viene dada por la necesidad del equipo de los servicios TIC de la EPSEVG, de tener más conocimiento sobre el uso que se les da a las aulas informáticas de la escuela. Debido a la gran cantidad de usuarios que ocupan estos equipos, los responsables de mantener estos sistemas funcionales, han solicitado una herramienta que les proporcione información, sobre el uso qué se les da a estas máquinas (siempre y cuando se encuentren conectadas a la red monitorizada).

La información recopilada por la herramienta, les podría ayudar con la planificación de nuevas distribuciones en las aulas informáticas de la escuela, proveyendo de ordenadores más potentes en las salas con más uso, dejando los equipos menos potentes y antiguos en las aulas menos usadas; así como evaluar el uso que se le da a los dos sistemas operativos ofrecidos en la escuela. Además, con esta información, se cuenta con la posibilidad de recibir avisos vía e-mail, sobre equipos que llevan determinado tiempo sin ser detectados en la red, pudiendo así, anticiparse a posibles quejas por parte de las personas que quisieran hacer uso de ellos.

Dado que la información se verá reflejada en una interfaz web amigable, se podrá observar en tiempo real el uso de los equipos conectados a las redes monitorizadas; permitiendo conocer si se encuentran conectados a la red e indicando cierta información para poder identificarlos.

3 OBJETIVOS

Vista la motivación de este proyecto, se pueden extraer los siguientes objetivos englobados en dos bloques:

- Desarrollo de una aplicación web minimalista y amigable que conste de las siguientes funcionalidades:
    - Creación de sistemas de escaneado personalizables, permitiendo al administrador definir la red a monitorizar, mediante el formato CIDR [1], y la frecuencia de ejecución del escaneado.
    - Detección de los equipos conectados a las redes monitorizadas, utilizando el protocolo ARP [2].
    - Detección del sistema operativo usado por los equipos previamente detectados, mediante el análisis de los servicios en ejecución en cada uno de ellos.
    - Visionado en tiempo real de los equipos pertenecientes a las redes monitorizadas, desde una sección dedicada en la interfaz web.
    - Clasificación de los equipos en grupos, permitiendo organizar los equipos de forma personalizada.
    - Cálculo de estadísticas de uso de los equipos en las redes monitorizadas, permitiendo escoger de que grupos de equipos se quiere analizar el uso.
    - Creación de dos tipos de alertas las cuales se notificarán vía email:
        - Detección de equipos activos en un rango de horas, permitiendo al administrador determinar el rango y en qué momento del día se va a recibir la notificación.
        - Detección de equipos inactivos, permitiendo al administrador determinar el número de días que debe estar un equipo sin detectarse, para que se adjunte a la notificación, y en qué momento del día se va a recibir el aviso.
- Despliegue de la herramienta en un servidor dedicado dentro de la universidad.
    - Análisis de los componentes requeridos
    - Integración de la aplicación web
    - Configuración de los componentes seleccionados

4 ESTADO DEL ARTE

4.1 Nagios

Nagios [3] es el sistema de monitorización de redes por excelencia. Es usado por gran parte de los administradores de todo el mundo y gracias a su condición de sistema de código abierto, bajo licencia GNU GPL V2, es mantenido por la comunidad.

Su mayor punto fuerte se encuentra en su filosofía de código abierto. Gracias a esto, la comunidad dispone de la capacidad de modificar Nagios a su antojo; dentro de los límites que permita la licencia. Así, los usuarios proporcionan distintos temas para la interfaz gráfica y multitud de plugins que añaden funcionalidades extra, no incluidas nativamente.

Mediante la especificación de los elementos que se quieren monitorizar, permite vigilar los equipos (hardware) y servicios (software), alertando cuando el comportamiento de los mismos no sea el deseado. Toda esta especificación se introduce manualmente para cada sistema a querer monitorizar. Además, si se quiere obtener información interna sobre los sistemas, se deben instalar ciertos programas en cada una de las máquinas, para que se comuniquen con el servidor en el que se encuentra instalado Nagios.

Está diseñado de tal forma, que permite responder de forma proactiva a los posibles problemas en los elementos especificados; reportándolos vía email y/o SMS, permitiendo al administrador solucionarlos incluso antes de que el cliente se dé cuenta.

Visto esto, parece la herramienta ideal para la monitorización de sistemas. Aun así, cuenta con el problema de que su configuración es muy tediosa, ya que se debe configurar cada sistema por separado. Su uso está enfocado a la monitorización de servidores. Por lo tanto, es poco viable configurarlo para los más de cien equipos disponibles en la escuela.

4.2 Icinga

Icinga [4] surgió como un fork (una variación) de Nagios. Sus características son muy similares a Nagios, dotándolo de la potencia y versatilidad de este. También cuenta con soporte para el uso los plugins desarrollados para Nagios.

Como en el caso anterior, su uso queda descartado al estar orientado a la monitorización de servidores y no a grandes cantidades de sistemas no tan críticos.

4.3 Nmap

Nmap [5] es una herramienta de código abierto para la exploración de red y auditoría de seguridad. Fue diseñado para el analizar rápidamente grandes redes, aunque funciona igual de bien analizando equipos individuales. Es capaz de detectar que equipos se encuentran disponibles en una red, que servicios y versiones ofrecen, que sistemas operativos ejecutan, que tipo de cortafuegos se está utilizando, entre muchas otras características. Aunque es ampliamente utilizado en auditorías de seguridad, muchos administradores de redes y sistemas lo encuentran útil para realizar tareas rutinarias, tales como la monitorización del tiempo que los equipos o servicios se mantienen activos. Se le conoce como la navaja suiza gracias a la gran cantidad de opciones y técnicas de escaneado que ofrece.

Si bien no se trata de ninguna herramienta de monitorización propiamente dicha, muchas de sus funcionalidades son útiles para desempeñar algunas de las tareas que se llevan a cabo. Más adelante en la memoria, se justificará su integración en este proyecto.

5 PLANIFICACIÓN

A continuación se documenta toda la información relacionada con la planificación del proyecto. El apartado de análisis de costes detalla los costes de todo el proyecto, separándolos en costes directos e indirectos. El apartado de paquetes de trabajo (Work Packages) detalla la agrupación por paquetes de las principales tareas planificadas del proyecto. En el aparatado del diagrama de Gantt se muestra la distribución de los paquetes de trabajo y sus tareas, a lo largo del periodo estimado de desarrollo del proyecto. Finalmente, se detallan los Milestones o puntos de control programados para el proyecto, así como las entregas planificadas.

5.1 ANÁLISIS DE COSTES

Este análisis de costes se hará partiendo de la hipótesis de que este proyecto, se trata de un proyecto profesional y no académico, con una duración de 20 semanas. Además se hará teniendo en cuenta que este es el único proyecto en el que se está trabajando y que por lo tanto los costes indirectos van al mismo y único cliente.

5.1.1 COSTES DIRECTOS

		Especificaciones	Coste € / semana
Equipo informático	Servidor	1 GB / 1 CPU 30 GB SSD	2,50 €
	Sistema operativo	Debian	Gratuito
	Aplicaciones	Open Source	Gratuito
Equipo de trabajo	Analista / Programador (Bruto)	Junior	550 €
	Seguridad Social	38,25%	210 €
Viajes y desplazamientos	Coche	Gasolina	15 €
	Tren	Billete	40 €
Total en € durante 20 semanas			**16.357,50 €**

Figura 5.1 Costes directos del proyecto

5.1.2 COSTES INDIRECTOS

	Detalle	Coste € / semana
Seguros	Responsabilidad profesional (pago único)	300 €
Estudio	Alquiler del estudio	150 €
	Luz, agua y gas	25 €
	Internet y teléfonos	15 €
Total en € durante 20 semanas		**4.100 €**

Figura 5.2 Costes indirectos del proyecto

5.1.3 COSTES TOTALES

Costes directos	16.357,50 €
Costes indirectos	4.100 €
Costes totales en € durante 20 semanas	**20.457,50 €**

Figura 5.3 Costes totales del proyecto

5.2 WORK PACKAGES

Seguidamente, se detallan los paquetes de trabajo en los que se han clasificado las tareas realizadas durante el transcurso de este proyecto.

WP0 – Gestión del proyecto

Semana de inicio 1 – Semana de finalización 20

Esfuerzo dedicado (persona/semana): 1

Tareas

- T0.1 - Revisión continua de la planificación y el estado del proyecto

WP1 – Diseño del sistema

Semana de inicio 1 – Semana de finalización 4

Esfuerzo dedicado (persona/semana): 1

Tareas

- T1.1 - Análisis de las necesidades
- T1.2 - Selección de los componentes

WP2 – Desarrollo de la aplicación

Semana de inicio 3 – Semana de finalización 17

Esfuerzo dedicado (persona/semana): 12

Tareas

- T2.1 - Desarrollo del módulo de escaneado
- T2.2 - Desarrollo del módulo de clasificación
- T2.3 - Desarrollo del módulo de estadísticas
- T2.4 - Desarrollo del módulo de alertas
- T2.5 - Implementación del módulo de autenticación y autorización
- T2.6 - Desarrollo de la interfaz web

WP3 – Despliegue

Semana de inicio 14 – Semana de finalización 18

Esfuerzo dedicado (persona/semana): 1

Tareas

- T3.1 - Instalación y configuración de los componentes
- T3.2 - Integración de la aplicación

WP4 – Validación

Semana de inicio 4 – Semana de finalización 17

Esfuerzo dedicado (persona/semana): 3

Tareas

- T4.1 - Testing del módulo de escaneado
- T4.2 - Testing del módulo de clasificación
- T4.3 - Testing del módulo de estadísticas
- T4.4 - Testing del módulo de alertas
- T4.5 - Testing de la aplicación en local, con datos simulados
- T4.3 - Testing de la aplicación en local, con datos reales
- T4.6 - Testing de la aplicación en entorno real

WP5 – Documentación

Semana de inicio 1 – Semana de finalización 20

Esfuerzo dedicado (persona/semana): 2

Tareas

- T5.1 - Redacción de la memoria escrita del proyecto
- T5.2 - Redacción del resumen del proyecto

5.3 MILESTONES

Milestone 1 – Módulo de escaneado

Fecha: Semana 6

Fin del desarrollo del módulo de escaneado.

Milestone 2 – Módulo de clasificación

Fecha: Semana 8

Fin del desarrollo del módulo de clasificación.

Milestone 3 – Módulo de estadísticas

Fecha: Semana 11

Fin del desarrollo del módulo de estadísticas.

Milestone 4 – Módulo de alertas

Fecha: Semana 13

Fin del desarrollo del módulo de alertas.

Milestone 5 – Interfaz web

Fecha: Semana 15

Fin del desarrollo de la interfaz web.

Milestone 6 - Aplicación en servidor dedicado

Fecha: Semana 18

Fin del despliegue de la aplicación en el servidor dedicado, integrándola con el resto de componentes.

5.4 DELIVERABLES

Deliverable 1 - Memoria del proyecto

Fecha: Semana 20

Entrega de la memoria escrita del proyecto.

Deliverable 2 - Resumen del proyecto

Fecha: Semana 20

Entrega del resumen del proyecto.

5.5 DIAGRAMA DE GANTT

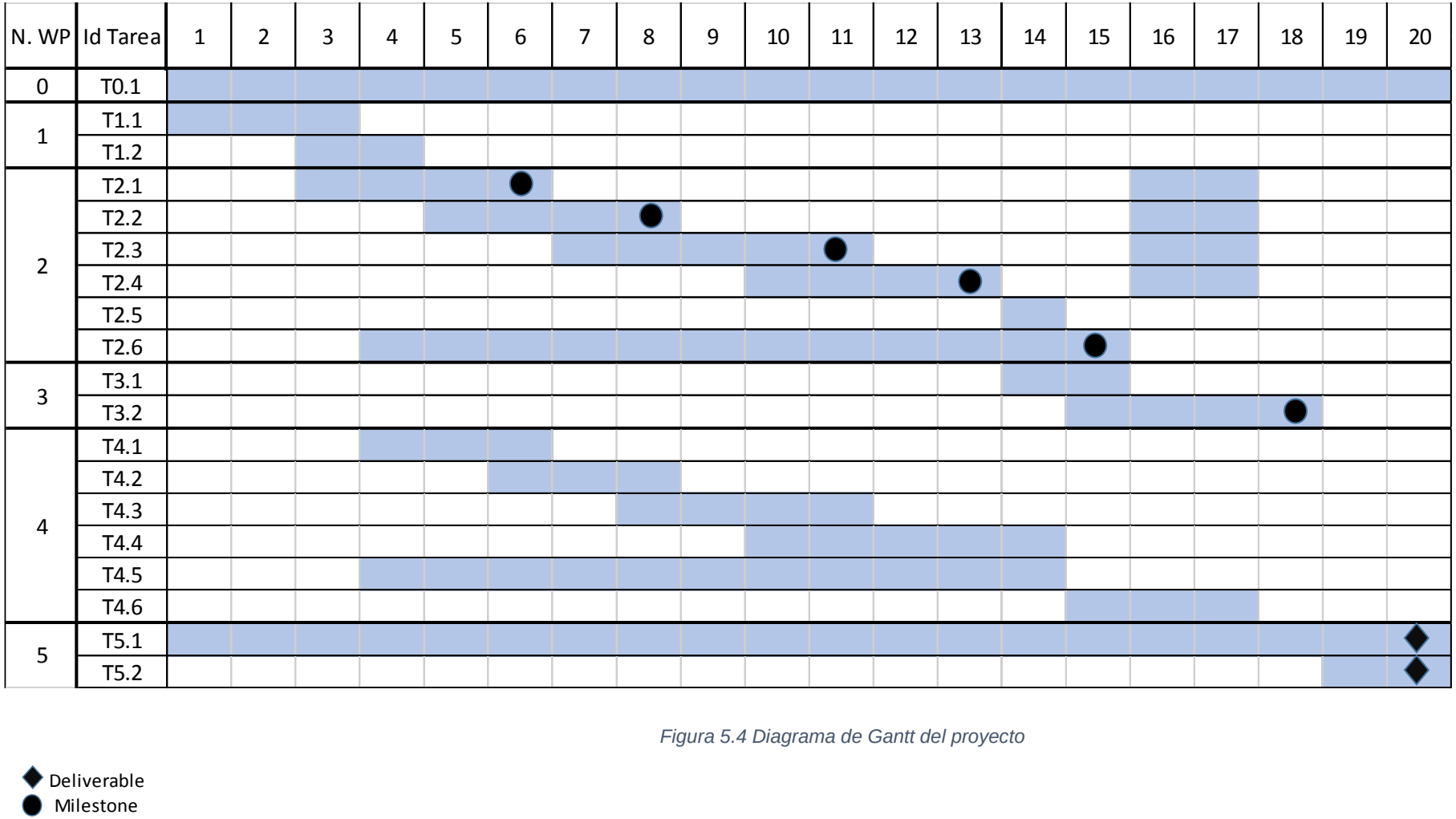


Figura 5.4 Diagrama de Gantt del proyecto

6 ANALISIS

A continuación se detallan las necesidades de las que consta este proyecto y la justificación de las elecciones llevadas a cabo.

6.1 Necesidades

Debido a la naturaleza de la herramienta desarrollada a lo largo del proyecto, es indispensable analizar las necesidades a tener en cuenta para su correcto funcionamiento.

6.1.1 Sistema Operativo

Dado que la aplicación va a estar viviendo dentro de una máquina virtual creada exclusivamente para ella en la EPSEVG; primeramente se debe escoger qué sistema operativo será el encargado de gestionar los recursos que se le proporcionarán.

El hecho de ser usado en la universidad, implica que cuanto menos impacto económico tenga hacia ella mejor, por lo que se descarta el uso de software propietario. Por lo tanto el sistema operativo que se usará, debe ser de software libre y gratuito.

El sistema operativo debe ser lo más liviano posible, con el fin de consumir los recursos únicamente indispensables para su funcionamiento. Esto implica prescindir de la típica interfaz gráfica que incluyen hoy en día la mayoría de sistemas operativos y otros servicios de los que no se va a hacer uso.

Finalmente, el sistema operativo a usar, debe contar con un buen soporte por parte de la comunidad y de sus desarrolladores, manteniéndolo al día frente a posibles amenazas.

6.1.2 Lenguaje de programación interpretado en el back-end

Hoy en día, la inmensa mayoría de aplicaciones web son dinámicas, esto quiere decir, que se permite al usuario interactuar con ella ofreciendo una experiencia más satisfactoria. Así mismo, la aplicación web planteada para este proyecto ofrecerá estas características. Para desempeñar esta tarea no basta con usar HTML y CSS, pues estos están muy limitados; se tienen que combinar con otro tipo de lenguajes de programación. Estos lenguajes son interpretados y ejecutados directamente tanto en el navegador del usuario (JavaScript), como en el servidor (Python, PHP).

En la aplicación web de este proyecto, la mayor parte del trabajo se va a desempeñar en el lado del servidor. Para la selección del lenguaje a utilizar, se tendrá en cuenta los siguientes aspectos:

- Popularidad. Es importante tener en cuenta la popularidad del lenguaje a seleccionar. Cuanto más popular sea, más librerías hechas por otros usuarios de la comunidad estarán disponibles. Gracias a esto, la aplicación final será más robusta al hacer uso de librerías que han sido testeadas por un gran número de personas, reduciendo los posibles bugs que pudieran surgir.
- Usabilidad. Al final del proyecto, se comentaran una serie de posibles mejoras de la aplicación desarrollada. Por ello, es importante tener en cuenta que el lenguaje utilizado para su desarrollo sea legible y fácil de entender, ofreciendo gran facilidad para añadir nuevas funcionalidades.
- Facilidad de aprendizaje. Antes de desarrollar este proyecto, no se había programado nunca una aplicación web, por lo que se desconocía tanto la forma de programar como su funcionamiento. Cómo el tiempo del proyecto es limitado, se valoró usar el lenguaje más intuitivo y con mayor rapidez de aprendizaje.
- Rapidez de ejecución. Al tener gran parte de la carga de trabajo en el servidor, es importante que el tiempo de ejecución de los procesos, sea lo más bajo posible. Con esto se conseguirán dos cosas, el usuario final disfrutará de una experiencia más fluida y el servidor no estará con una carga de trabajo constante.

6.1.3 Framework

Existen una serie de tareas realizadas en el desarrollo web, las cuales se repiten constantemente. Un framework permite aligerar la carga de trabajo del desarrollador, permitiendo abstraerle de esas tareas asociadas al desarrollo web. Por ejemplo, la mayoría de los frameworks disponibles cuentan con librerías para el acceso a la base de datos, motores para las plantillas y cuentan con un sistema de gestión de sesiones. Además, promueven la reusabilidad del código, permitiendo al desarrollador no repetirse para cada aplicación que desarrolle.

Para este proyecto se pretende usar un framework que siga el patrón de desarrollo MVC [6], Modelo-Vista-Controlador. Este patrón separa el modelo de datos de la lógica de negocio y de la interfaz de usuario. Esto permite programar de forma modular, promoviendo el principio DRY, don’t repeat yourself.

6.1.4 Sistema de gestión de bases de datos

Debido a la naturaleza de la herramienta desarrollada en este proyecto, es necesario almacenar grandes cantidades de datos con el fin de poder analizarlos y ofrecer información útil sobre ellos. Dicho esto, queda claro que es necesario almacenar los datos en una base de datos, ya que no es viable almacenarlos en ficheros planos.

Dada la naturaleza de los datos que se obtendrán, se hace uso de un sistema gestor de bases de datos relacional; dejando de lado las bases de datos no relacionales. Como ocurre con el resto de necesidades, es importante que el sistema utilizado sea gratuito.

6.1.5 Servidor web

Ya que la aplicación que se desarrolla es una aplicación web, se debe poder recibir peticiones HTTP de parte de los clientes que quieran hacer uso de ella. Para eso es necesario disponer de un software que se encargue de gestionar las conexiones entre el cliente y el servidor, aceptando o rechazando las peticiones HTTP, entregando las páginas web solicitadas, etc.

Puesto que el uso que se le va a dar a esta aplicación, a nivel de conexiones recibidas, será muy limitado (únicamente los administradores de la EPSEVG dispondrán de acceso a la herramienta), se busca un servidor web ligero y a la vez rápido.

6.1.6 Aplicaciones de escaneado

Una parte de la aplicación se dedica a la recopilación de información sobre los dispositivos que se encuentran conectados a las redes monitorizadas. A continuación se especifica que información se desea obtener:

- Detección de los dispositivos activos y conectados a la red. Ya que la aplicación está pensada para monitorizar redes de área local, en las que debe estar presente el servidor, la detección de los equipos se realizará mediante peticiones ARP.
- Detección del sistema operativo. La detección del sistema operativo se hará en base a las direcciones IP detectadas en el escaneado anterior. Se analizará una serie de servicios característicos de los sistemas operativos más conocidos, con tal de identificarlos.
- Detección del nombre de los dispositivos. Mediante consultas al DNS local, se obtendrán los nombres de los dispositivos de la red.

6.2 Elecciones

En este apartado, se detallan las elecciones efectuadas en base a las necesidades indicadas en el apartado anterior.

6.2.1 Sistema Operativo

Existen diversas distribuciones de GNU/Linux que cumplen las necesidades mencionadas. Pero hay dos que resaltan por encima de los otros. Estos son Debian [7] y CentOS [8], ambas en sus versiones de servidor.

Tanto Debian como CentOS son muy utilizados en servidores a nivel mundial. Ambas distribuciones son muy estables y se mantienen al día con las actualizaciones para parchear amenazas de seguridad.

Se escoge Debian como sistema operativo para este proyecto, simplemente por la experiencia previa de administración en esta distribución.

6.2.2 Lenguaje de programación interpretado en el back-end

Siguiendo los valores mostrados en la página web TIOBE [9], dedicada en parte a medir la popularidad de todos los lenguajes de programación existentes, se extraen tres de los lenguajes más populares para este tipo de aplicaciones PHP, Python y Ruby. Todos ellos cuentan con un gran soporte por parte de la comunidad, pero tal y como se muestra la Figura 6.1, PHP (al ser el más antiguo) resalta por encima de los otros dos.

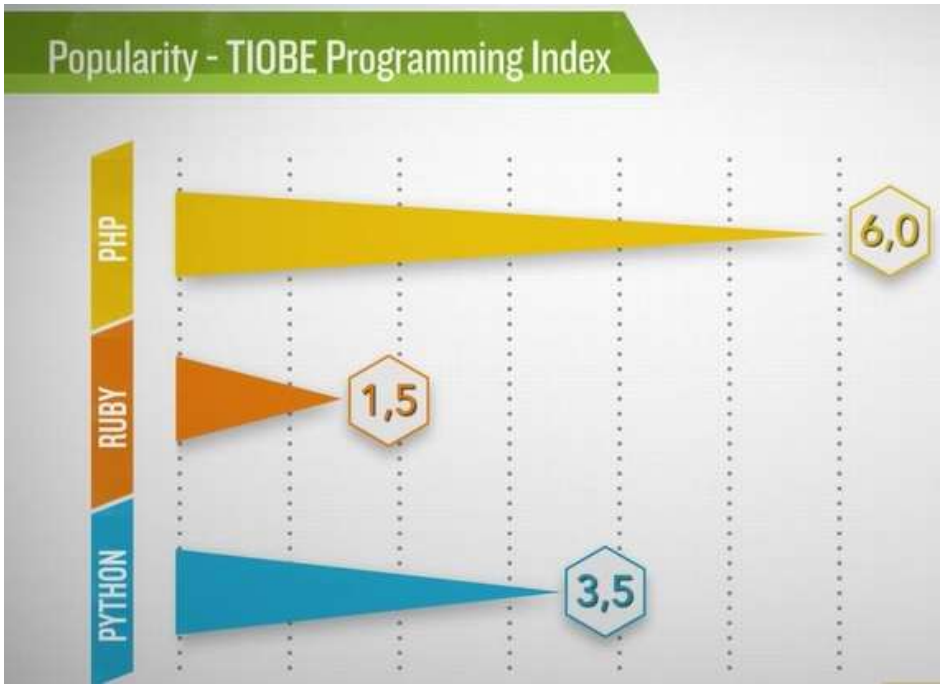


Figura 6.1 Índice de popularidad de los tres lenguajes de back-end más usados

Tal y como se muestra en la Figura 6.2, en temas de usabilidad gana Ruby, seguido de Python.

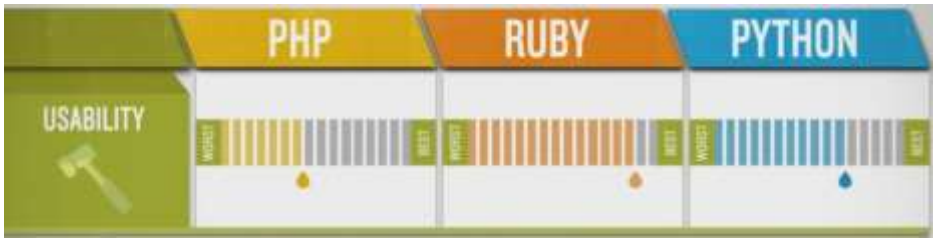


Figura 6.2 Niveles de usabilidad de los tres lenguajes de back-end más usados

En la Figura 6.3 se muestra que el lenguaje más fácil de aprender se trata de Python, esto es así ya que tanto su sintaxis como su estructura, se asimilan de cierta forma al lenguaje natural.

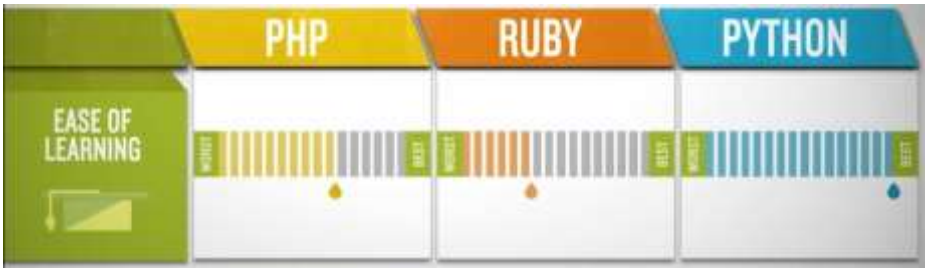


Figura 6.3 Niveles de facilidad de aprendizaje de los tres lenguajes de back-end más usados

Finalmente, en la Figura 6.4 se expone por un lado, una comparativa del número de líneas de código, necesarias para programar una misma tarea en cada uno de los tres lenguajes. Por el otro lado, se muestra la rapidez de ejecución de esa misma tarea en cada uno de los lenguajes. Se observa que en número de líneas de código PHP y Python están a la par, pero en rapidez de ejecución Python es claramente el vencedor.

Figura 6.4 Número de líneas de código y tiempo de ejecución para una misma tarea

Las anteriores figuras mostradas, han sido extraídas del blog udemy [10].

A raíz de estas comparativas, se ha seleccionado Python, en concreto su versión 3.4.2, como lenguaje de programación por su fácil curva de aprendizaje y su rapidez de ejecución.

6.2.3 Framework

El framework a usar está fuertemente ligado al lenguaje de programación que se usará en el servidor. Existen varios frameworks web disponibles para Python [11]; los más conocidos son Django, TurboGears y web2py. Estos son los más completos ya que integran gran parte de las funcionalidades mencionadas en el apartado de necesidades.

Se descarta el uso de TurboGears, al contar este con muchas dependencias para su funcionamiento, condicionándolo a posibles incompatibilidades entre versiones. También se descarta el uso de web2py al no ser totalmente compatible con la versión 3 de Python.

Así, en este proyecto se ha usado Django [12], en su versión 1.7, como framework web puesto que sigue la filosofía anteriormente citada, contando además, con sus propios componentes, solucionando los posibles problemas de compatibilidades.

6.2.4 Sistema de gestión de bases de datos

Los SGBD relacionales gratuitos más conocidos son SQLite, MySQL, MariaDB y PostgreSQL.

Se descarta el uso de SQLite por dos razones. La primera de ellas es que no es una base de datos como tal, se almacena en forma de fichero de datos debidamente organizados y esto incumple una de las condiciones impuestas en las necesidades. La segunda razón y la más importante por la que se descarta es que sufre problemas de rendimiento cuando trabaja con grandes cantidades de datos.

Las tres opciones que restan son muy buenas candidatas, pero tanto MySQL y MariaDB (fork de MySQL), cuentan con un problema un tanto peculiar. Estos dos sistemas no cumplen con algunos de los estándares establecidos de SQL. Aun así son ampliamente utilizados mundialmente.

Se ha escogido PostgreSQL [13] como SGDB puesto que se recomienda su uso en la documentación de Django [14], ofreciendo el mejor rendimiento.

6.2.5 Servidor web

Se han tenido en cuenta dos servidores web para llevar a cabo la tarea planteada, estos son Apache y Nginx. Ambos son ampliamente conocidos, son robustos y se desenvuelven con soltura tanto en webs con mucha demanda como en webs pequeñas.

Se ha seleccionado Nginx [15] para desempeñar esta tarea, teniendo en cuenta el servidor web recomendado en la documentación de Django.

Para poder comunicar la aplicación desarrollada en Python con el servidor web, es necesario usar una interfaz que se encargue del intercambio de información entre estos. En Python se conoce como Web Server Gateway Interface (WSGI) [16]. Para el proyecto, se ha escogido como en casos anteriores, la interfaz recomendada en la documentación del Django, esta es Gunicorn [17].

6.2.6 Aplicaciones de escaneado

Detección de los dispositivos activos y conectados a la red

Existen varias aplicaciones capaces de realizar la detección de los equipos mediante consultas ARP. Algunas de ellas son Arpscan y Nmap. La primera se dedica exclusivamente al envío de paquetes ARP a todos los sistemas de la red local, mostrando si se recibe algún tipo de respuesta o no. Si hay respuesta, significa que el equipo se encuentra activo.

Nmap por el otro lado, es una herramienta muchísimo más potente, como ya se ha comentado en el Capítulo 4. Una de las opciones que incorpora, es idéntica en funcionamiento a Arpscan, devolviendo la correspondencia MAC – IP de los equipos.

Detección del sistema operativo

Para desempeñar esta tarea, se podrían utilizar una serie de herramientas como Amap, Dmitry, Xprobe2, entre otras; encargadas de detectar los servicios de los sistemas activos de la red e intentar averiguar de forma poco fiable el sistema operativo que hay detrás.

Para este tipo de detecciones Nmap es la mejor opción. Al realizar esta tarea, Nmap envía paquetes TCP y UDP a los equipos remotos y examina prácticamente cada bit de las respuestas. En base a las respuestas obtenidas, coteja la gran base de datos de la que dispone para determinar con la máxima precisión posible de que sistema operativo se trata.

Detección del nombre de los dispositivos

DNSdict6, DNSenum, DNSmap y Reverseraider son algunas de las herramientas dedicadas al análisis de DNS. Todas ellas realizarían la tarea de forma impecable. Pero como en los casos anteriores, Nmap sobresale en su eficiencia en la ejecución, permitiendo por defecto, realizar peticiones inversas al DNS configurado en el propio servidor y así obtener los nombres de todas las máquinas activas registradas en él, conociendo únicamente sus direcciones IP.

Vistos los tres tipos de detecciones que se requieren y las herramientas capaces de desempeñar esas funciones; en este proyecto se hará uso de Nmap, siendo la única herramienta de todas ellas que engloba los tres tipos de detecciones.

6.3 Arquitectura del sistema

A continuación, en la Figura 6.5 se muestra un esquema indicando cómo están relacionados los distintos componentes definidos anteriormente, excluyendo Nmap pues no afecta al funcionamiento del servidor. Se puede observar como Nginx es el encargado de ofrecer visibilidad del servidor hacia el exterior, pudiendo encontrarse las redes locales e internet. PostgreSQL podría estar situado tanto dentro como fuera del servidor puesto que ofrece la posibilidad de conexión remota; en este proyecto se encuentra en el interior del servidor.

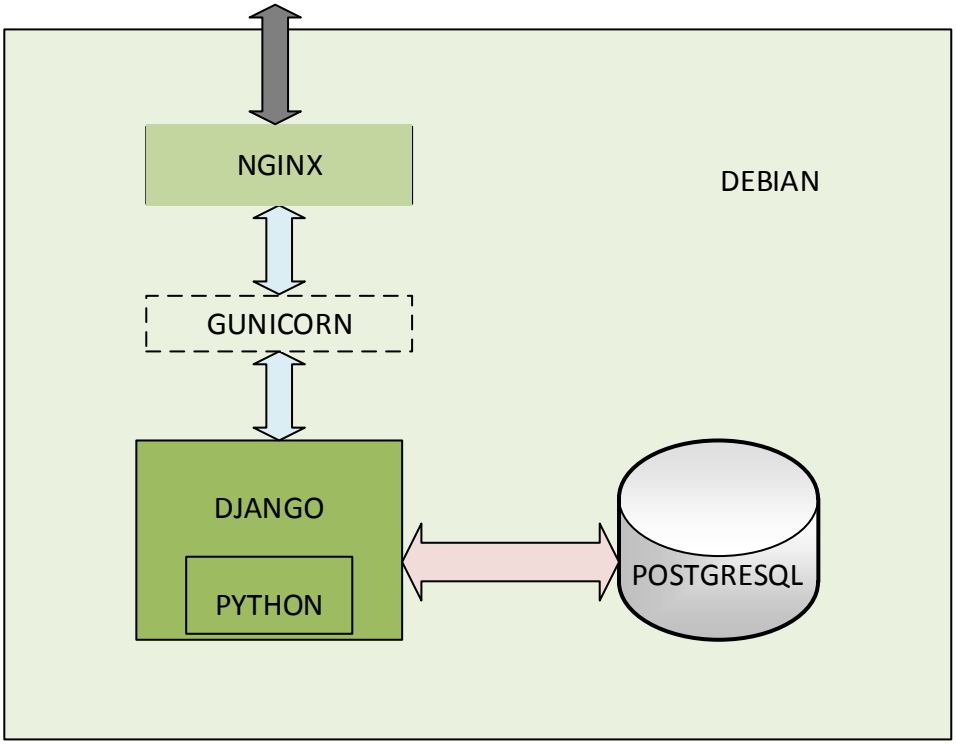


Figura 6.5 Relaciones entre los diferentes componentes seleccionados para el proyecto

7. DISEÑO DE LA INTERFAZ WEB

Dada la naturaleza de la aplicación desarrollada, el administrador interactúa con ella mediante dos interfaces web propiamente desarrolladas. A continuación se detallará cuáles son estas interfaces, qué información se muestra en cada una de ellas y cuál es su propósito.

7.1 Tipos de interfaz

Como se acaba de mencionar, la aplicación web consta de dos interfaces distintas, pero a la vez relacionadas entre ellas. La primera consta de un diseño minimalista y amigable, que permite tener una visión muy rápida, sobre los escáneres configurados y sobre el estado de todos los equipos detectados. Esta es la interfaz que se observa por defecto al acceder a la aplicación. En el desarrollo de esta interfaz, se ha usado Bootstrap [20] para poder ofrecer un diseño responsive, pudiéndose adaptar a todo tipo de pantallas. Estas vistas, contienen un contador en el código HTML que las refresca cada 5 minutos, pudiendo ver los cambios efectuados en la base de datos.

La segunda interfaz cuenta con un diseño más complejo, para que el administrador pueda interactuar con todos los módulos desarrollados. Es en esta interfaz, dónde se puede exprimir a fondo la aplicación, permitiendo a los administradores disponer de todos los datos y configuraciones generadas para la aplicación web. Cuenta con las opciones de ordenación, búsqueda y filtrado ofreciendo una experiencia más practica al administrador. Esta interfaz se detallará en el siguiente capítulo, pues su uso se entenderá mejor, explicando cada uno de sus apartados de forma separada en cada uno de los módulos desarrollados.

Cabe destacar que ambas interfaces se encuentran en inglés. Se ha tomado esta decisión porqué la mayoría de herramientas de esta índole se encuentran en este idioma.

7.2 Vistas de la interfaz minimalista

En este apartado se detallarán las vistas que ofrece la interfaz web minimalista desarrollada para este proyecto. Para cada una de ellas, se informarán de ciertos puntos a destacar.

Las imágenes añadidas en los siguientes puntos, han sido tomadas en distintos momentos del día. Por lo que la información en cada una de ella puede variar.

7.2.1 Página de inicio de sesión

Esta es la primera vista que se observa al acceder a la web. Para poder ofrecer privacidad de los datos recopilados, la aplicación consta de una pantalla de inicio de sesión. Ya que durante el desarrollo del proyecto, se han estado haciendo pruebas en remoto con el servidor ubicado en la EPSEVG, se ha implementado este sistema de login para poder proteger la información interna de la red de la escuela.

Como se puede observar en Figura 7.1, se solicitan unas credenciales para poder acceder a la aplicación y poder ver su contenido.

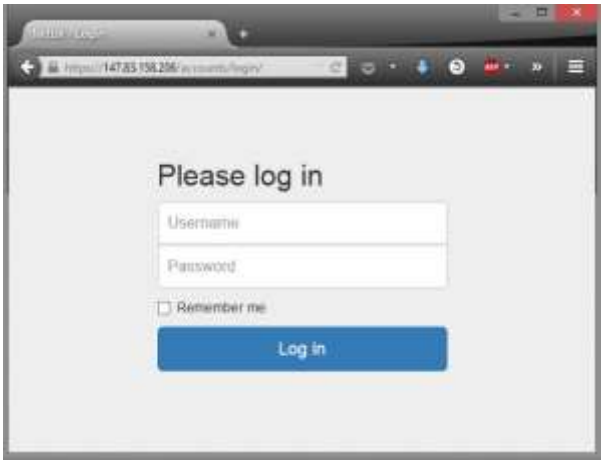


Figura 7.1 Pantalla de inicio de sesión de la aplicación

7.2.2 Página principal

Cuando se ha realizado con éxito el login, frente al administrador, aparecerá una pantalla con todos los escáneres configurados hasta ese momento. Esta vista se analizará siguiendo la numeración dispuesta en la Figura 7.2.

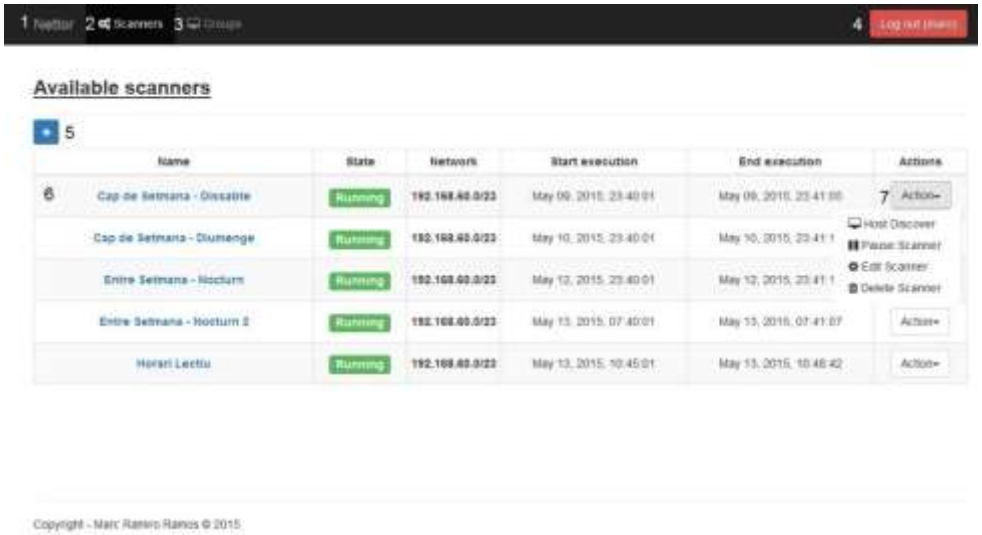


Figura 7.2 Página principal de la aplicación

1. **Acceso a la interfaz de administración.** Este enlace está situado en la barra de navegación, presente en toda la interfaz minimalista. Pulsando en *Nettor* se accede a la interfaz de administración.
2. **Acceso a la vista de escáneres disponibles.** Pulsando en *Scanners,* se accede a la pantalla principal, donde se muestra el listado de todos los escáneres disponibles.
3. **Acceso a la vista de grupos.** Pulsando en *Groups,* se accede a la pantalla donde se encuentran todos los grupos de equipos detectados por la aplicación.
4. **Logout de la aplicación.** Este botón muestra el usuario que está haciendo uso de la aplicación. Cuando se pulsa, se procede al cierre de sesión del usuario.
5. **Añadir nuevo escáner.** Este botón conduce al administrador a la vista de creación de escáneres disponible en la interfaz de administración.
6. **Tabla con los escáneres disponibles.** En esta tabla, se muestran por filas, todos los escáneres que se encuentran configurados en la aplicación. Se muestran los siguientes campos:
    a. **Name.** Nombre dado al escáner. Pulsando en el enlace, se redirige la vista a la interfaz de administración, para la modificación del escáner.
    b. **State.** Estado en el que se encuentra el escáner. Se dará más información en el próximo capítulo.
    c. **Network.** Red monitorizada por el escáner.
    d. **Start execution.** Inició de la última ejecución del escáner.
    e. **Ens execution.** Fin de la última ejecución del escáner.
    f. **Actions.** Se explica a continuación.
7. **Acciones disponibles.** En este desplegable se muestra una serie de acciones disponibles para cada escáner. A continuación se indica el propósito de cada una de ellas:
    a. **Host discover.** Permite la ejecución de un único escaneo a la red definida.
    b. **Pause Scanner.** Esta acción cambia el estado del escáner a *Paused.*
    c. **Edit Scanner.** Redirige la vista a la interfaz de administración para la modificación del escáner.
    d. **Delete Scanner.** Elimina el escáner.

7.2.3 Página de grupos

En esta página, se muestra un listado de todos los grupos detectados en las redes escaneadas y los grupos creados manualmente. Esta pantalla se puede ver en la Figura 7.3.

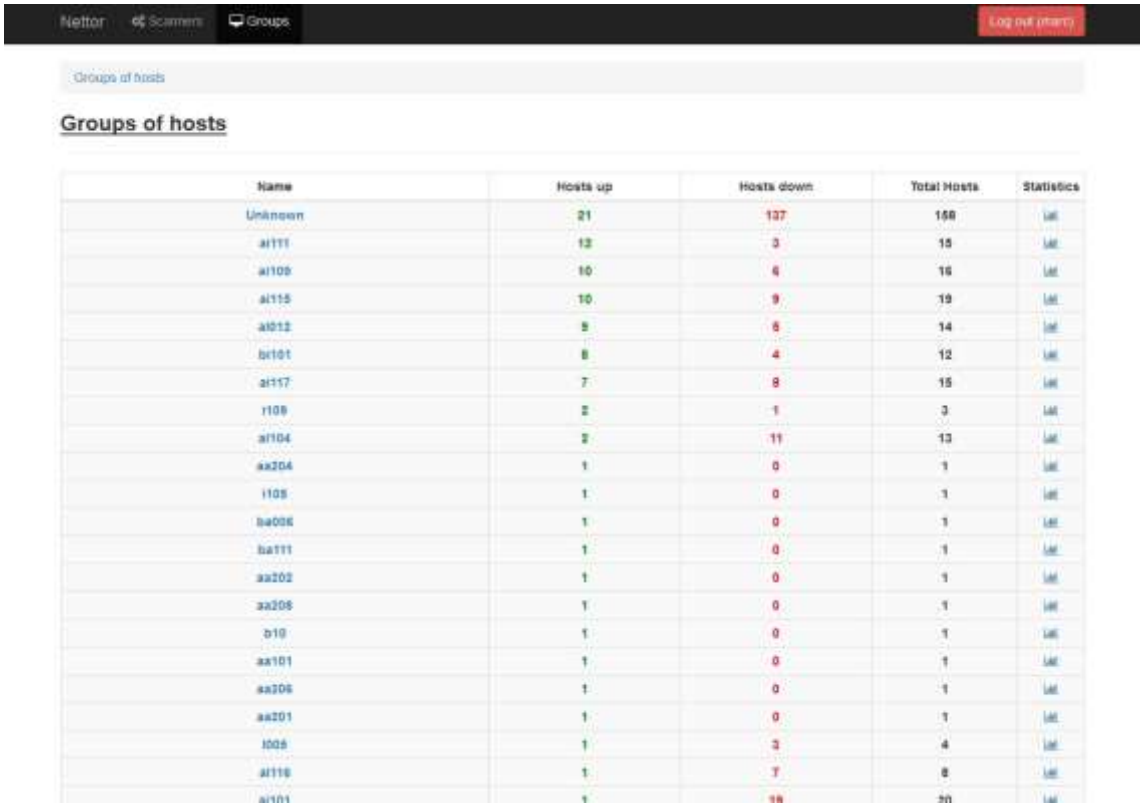


Figura 7.3 Listado de todos los grupos detectados

A continuación se describe el significado de cada uno de los campos:

1. **Name**. Nombre del grupo. Este puede ser en base a los equipos que pertenecen a él o simplemente es el nombre descriptivo que se le ha dado manualmente. El grupo *Unknown*, es el grupo por defecto, donde se añaden los equipos que no se han clasificado automáticamente. Pulsando en el nombre, se redirige página a la pantalla de equipos.
2. **Hosts up**. Este valor indica el número de equipos detectados por el último escaneo realizado.
3. **Hosts down**. Indica el número de equipos no detectados por el último escaneo realizado.
4. **Total Hosts**. Este valor indica la cantidad de sistemas que pertenecen a cada grupo.
5. **Statistiscs**. Este enlace, redirige la vista a la interfaz de administración, para el visionado de estadísticas diarias del grupo.

7.2.4 Página de equipos

En esta página, se muestra un listado de todos los equipos detectados, clasificados por su grupo, en las redes escaneadas. Esta página se analizará siguiendo la numeración dispuesta en la Figura 7.4.

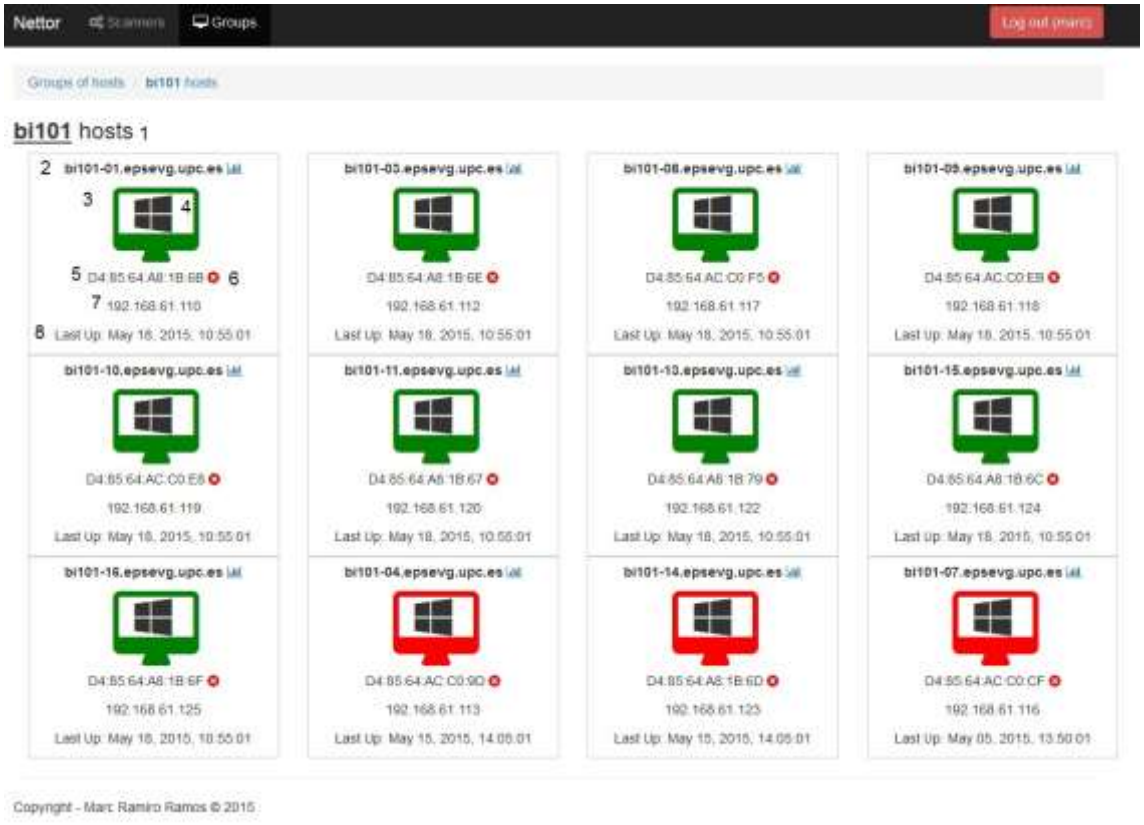


Figura 7.4 Listado de los equipos detectados del grupo bi101

1. **Nombre del grupo** al que pertenecen los equipos mostrados.
2. **Nombre del equipo.** A su lado se dispone de un enlace a la vista de estadísticas diarias de este.
3. **Indicador del estado del equipo**. Si su color es verde, indica que el sistema se ha detectado durante ejecución del último escaneo. Por el contrario, si se encuentra rojo, indica que el sistema no ha sido detectado.
4. **Sistema operativo** que se ha detectado en ese equipo, durante último escaneado ejecutado dónde ha sido detectado.
5. **Dirección MAC** del equipo.
6. Botón para enviar el equipo a la **lista negra.**
7. **Dirección IP** del equipo, detectada durante el último escaneado ejecutado.
8. **Fecha de última detección** del equipo.

7.2.5 Página de equipos en la lista negra

En la Figura 7.5 se puede observar que al final de la lista de grupos, se encuentra un botón llamado *Blacklist*.

Figura 7.5 Botón de acceso a la lista negra

Cuándo el anterior botón es pulsado, este redirige la página a una lista con todos los sistemas añadidos a la lista negra, tal y como se observa en la Figura 7.6. El porqué de su uso, se documentará en el siguiente capítulo.

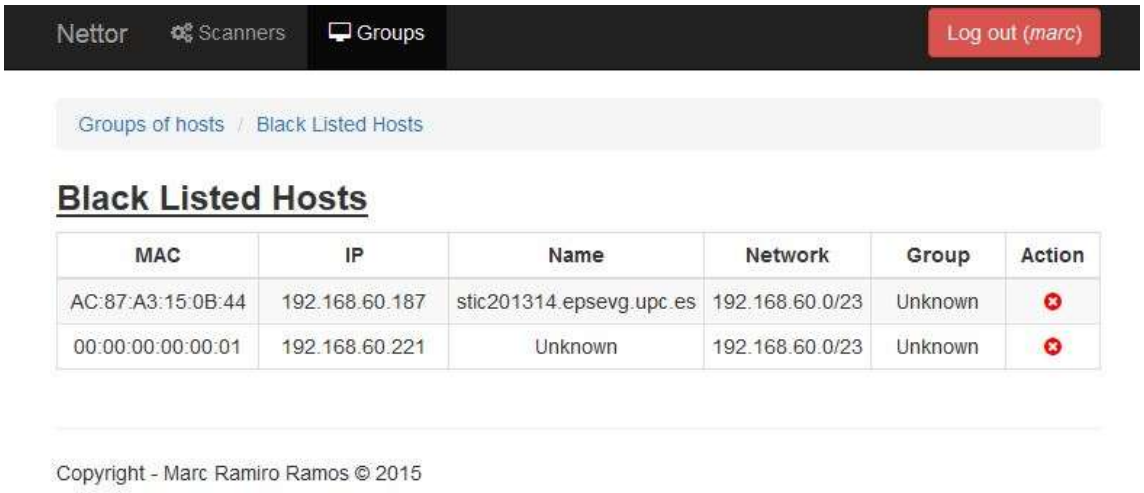


Figura 7.6 Sistemas incluidos a la lista negra

Para cada uno de sistemas en la lista, se ofrece la última información almacenada. La cruz roja presente en la columna *Action* permite eliminar los equipos de la lista negra, devolviéndolos a su anterior grupo.

8 IMPLEMENTACIÓN

En este capítulo se desarrollará el proceso de creación de la aplicación web y los mecanismos que utiliza. Se ofrece una explicación sobre cuatro módulos desarrollados a lo largo del proyecto. Finalmente, se muestra el diagrama de clases que relaciona los módulos entre ellos.

8.1 Módulos desarrollados

Siguiendo la filosofía de Django, la aplicación web se ha implementado de forma modular. Así, se han desarrollado cuatro módulos diferenciados por las tareas que desempeñan.

En el capítulo anterior, se ha hecho mención de la interfaz de administración de la que dispone la aplicación. En las siguientes explicaciones se mostrarán capturas de esta interfaz, complementando así la información dada.

En la Figura 8.1 se muestra la página inicial de la interfaz de administración, dónde se pueden observar los módulos desarrollados.

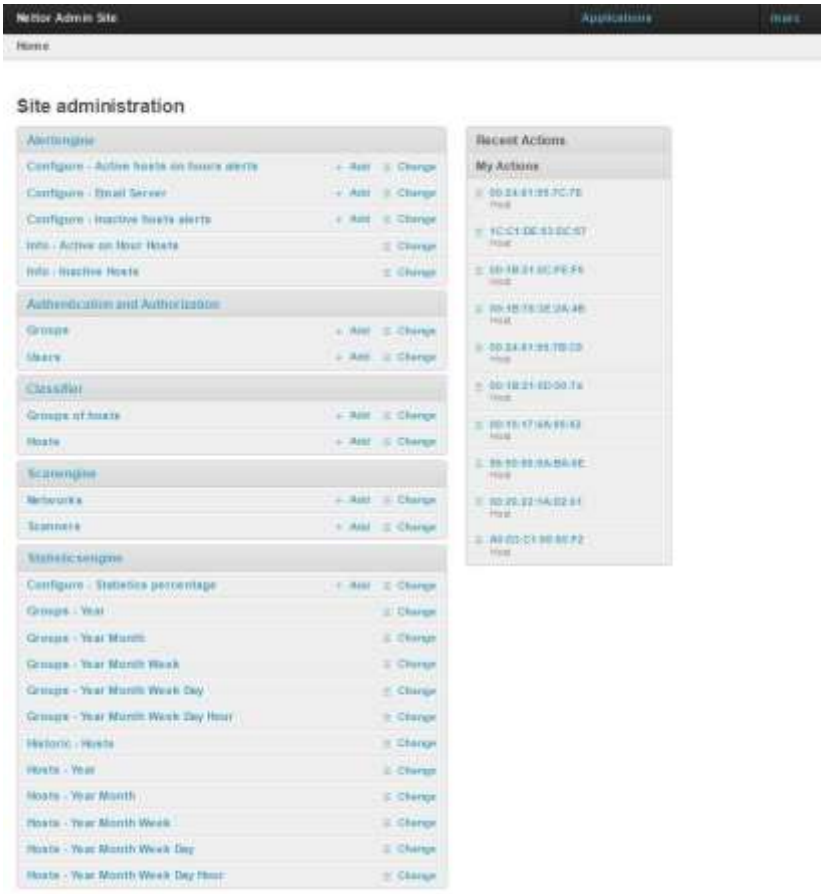


Figura 8.1 Página inicial de la interfaz de administración

8.1.1 Módulo de escaneado

La base del funcionamiento de la aplicación desarrollada en este proyecto, se encuentra en las detecciones expuestas en el capítulo 6. Para poder realizar estas detecciones, este módulo consta de dos partes.

La primera parte, ofrece al administrador la capacidad de programar escaneados personalizados. Esto quiere decir, que se le da la posibilidad de indicar qué red se desea monitorizar y cada cuanto tiempo se debe ejecutar el escaneado. La red especificada debe estar en el formato CIDR, permitiendo monitorizar grandes cantidades de equipos, sin tener que especificar sus direcciones IP una a una. El tiempo especificado, permite indicar que días de la semana y durante que rango de horas se desea monitorizar la red.

Finalmente, para cada escáner creado, se ofrece la opción de registrar los datos obtenidos en un histórico, para su posterior análisis y generación de estadísticas; esto se detallará en los próximos apartados.

En la Figura 8.2 se muestra la visión del administrador al configurar un nuevo escáner.

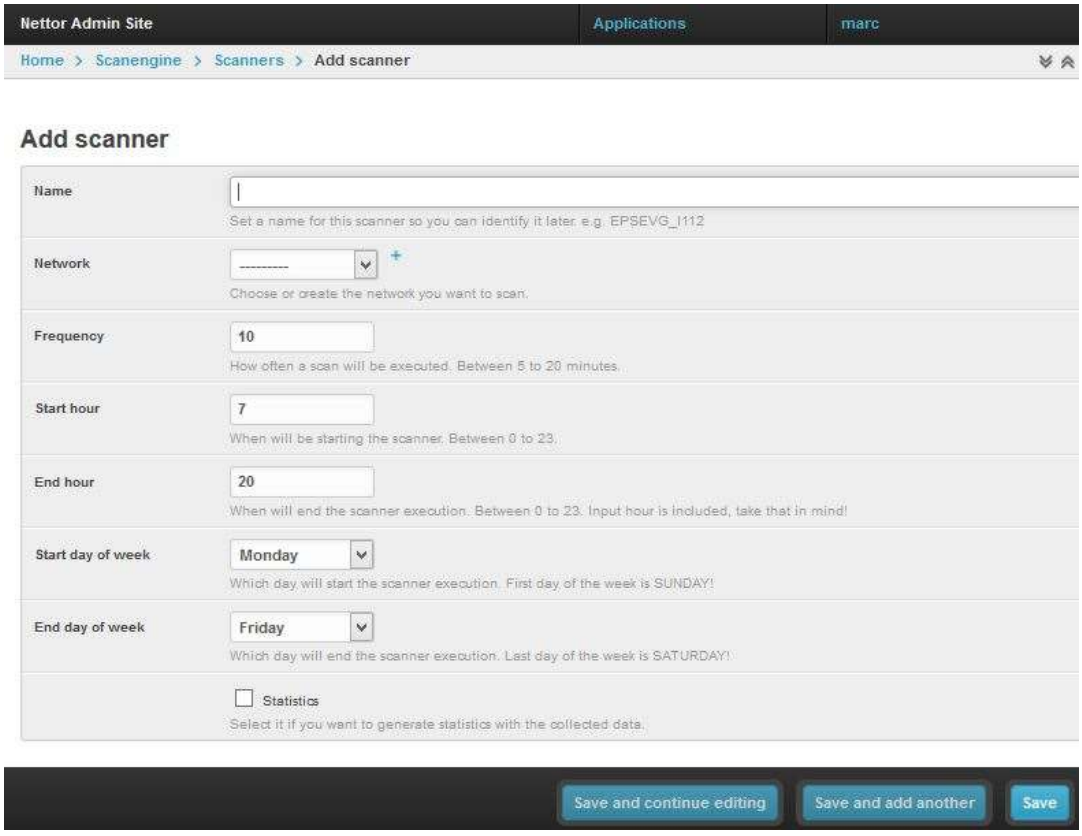


Figura 8.2 Pantalla de creación de un nuevo escáner

Una vez creado el escáner, se podrá consultar, modificar y eliminar en cualquier momento, tal y como se puede observar en la Figura 8.3 y la Figura 8.4.

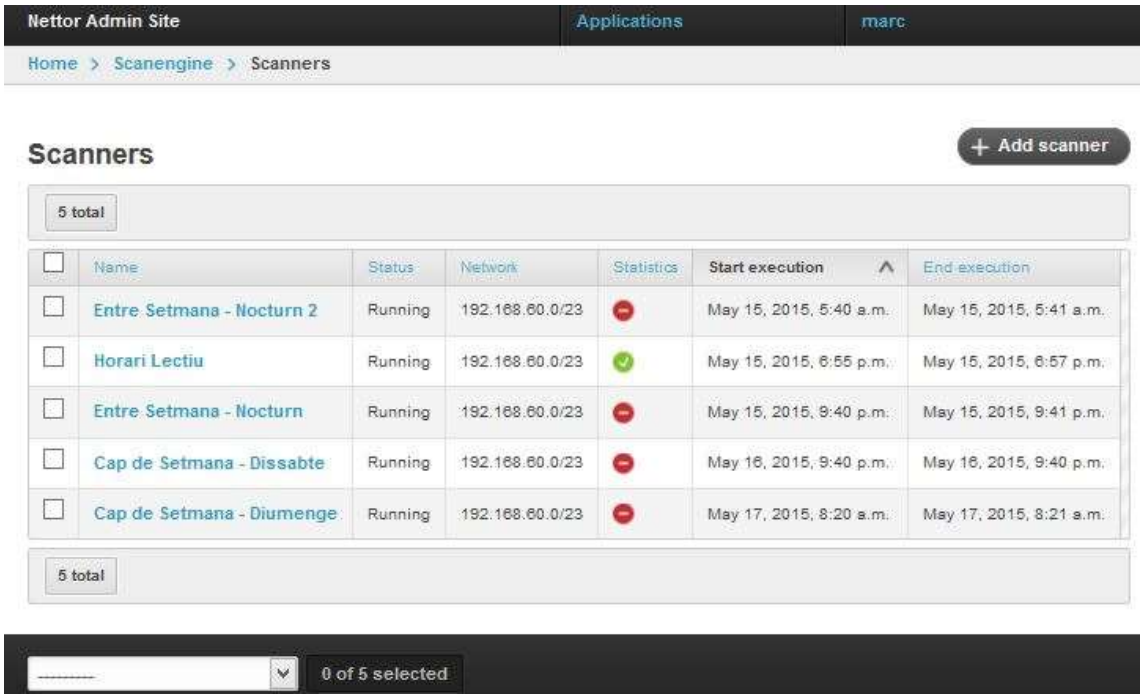


Figura 8.3 Listado de escáneres creados

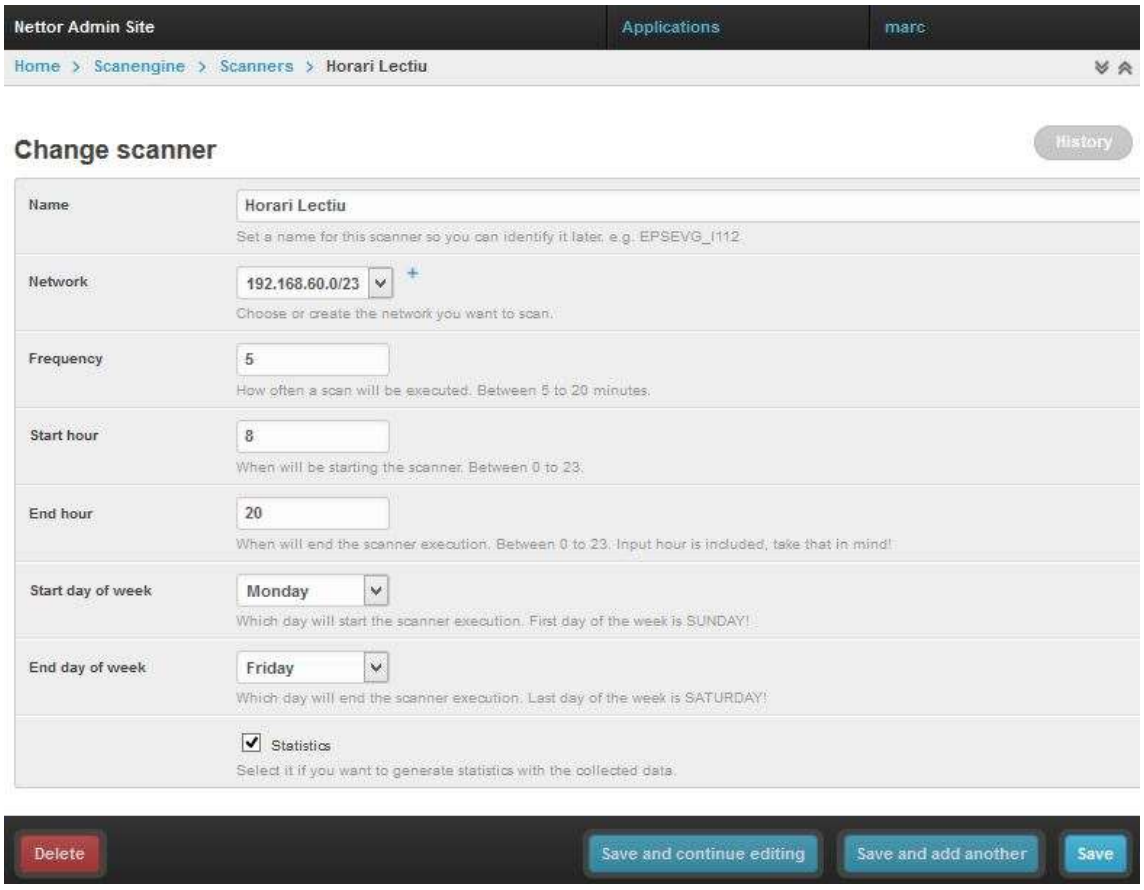


Figura 8.4 Consultando la configuración de uno de los escáneres existentes

La segunda parte, consta de los procesos internos de la aplicación, encargados de ejecutar el escaneado cómo y cuándo toca. Para ello, hay un indicador del estado en que se encuentran. Existen tres estados disponibles:

- Running. Indica que se encuentra operativo.
- Paused. Indica que el escáner se encuentra pausado de forma indefinida.
- Stopped. Indica que se trata de un nuevo escáner, sin previa activación.

Cuándo un escáner se activa y por lo tanto pasa al estado *Running*, la aplicación añade una nueva entrada en el cron [18] del servidor, con la configuración temporal del escáner introducida previamente. Una vez se cumple la condición del tiempo en el cron, este ejecuta una serie de scripts que interactúan con Nmap y el resto de los módulos, analizando la red definida en el escáner.

El primer script ejecutado, se encarga de detectar los dispositivos activos de la red en ese instante, almacenando por un lado la información de estos y por el otro, almacenando en un fichero, un listado de todas las IPs detectadas. Además, dentro de la misma ejecución de Nmap, se hacen las consultas inversas al DNS para obtener el nombre de los equipos.

El segundo script toma el fichero de IPs de entrada y ejecuta la detección del sistema operativo sobre ellas, almacenando el resultado en un fichero.

8.1.1.1 Diagramas de flujo

A continuación se muestran los diagramas de flujo que sigue este módulo:

Figura 8.5 Creación de un escáner

Figura 8.6 Modificación de un escáner

Figura 8.7 Eliminación de un escáner

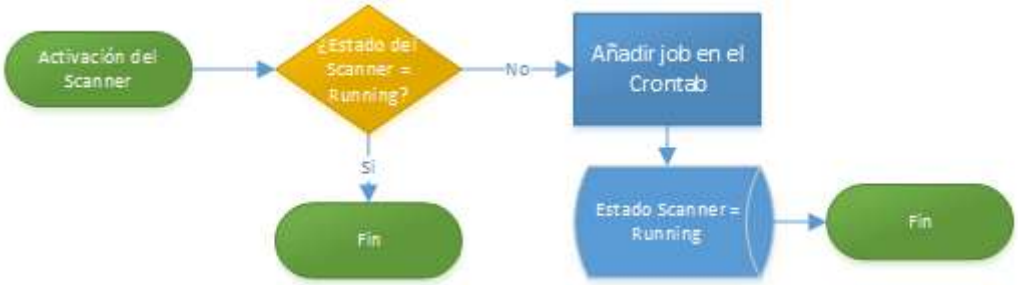


Figura 8.8 Activación de un escáner

Figura 8.9 Pausado de un escáner

Figura 8.10 Ejecución de un escáner

8.1.2 Módulo de clasificación

Como se ha indicado en el apartado anterior, al finalizar la ejecución de los scripts que interactúan con Nmap, se crean dos ficheros distintos. El primero de ellos incluye la información básica de los equipos, indicando su nombre (en el caso de que este registrado en el servidor DNS), dirección MAC y dirección IP. En la Figura 8.11 se puede observar un extracto de este fichero.

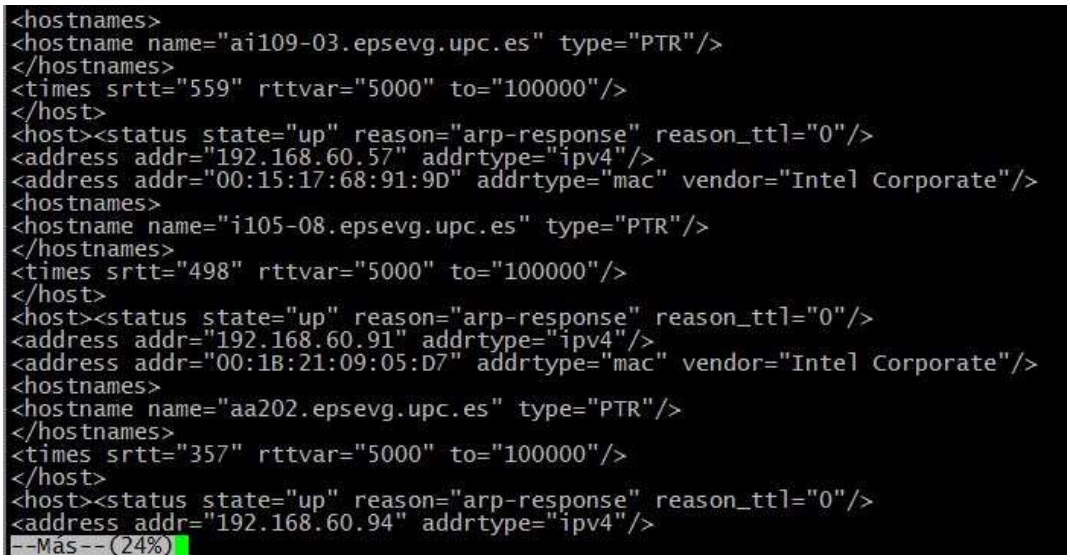


Figura 8.11 Extracto del fichero de respuesta de Nmap, para la detección de equipos

El segundo de ellos obvia la resolución inversa de nombres y únicamente contiene las direcciones MAC e IP y el sistema operativo. En la Figura 8.12 se puede observar un extracto de este otro fichero.

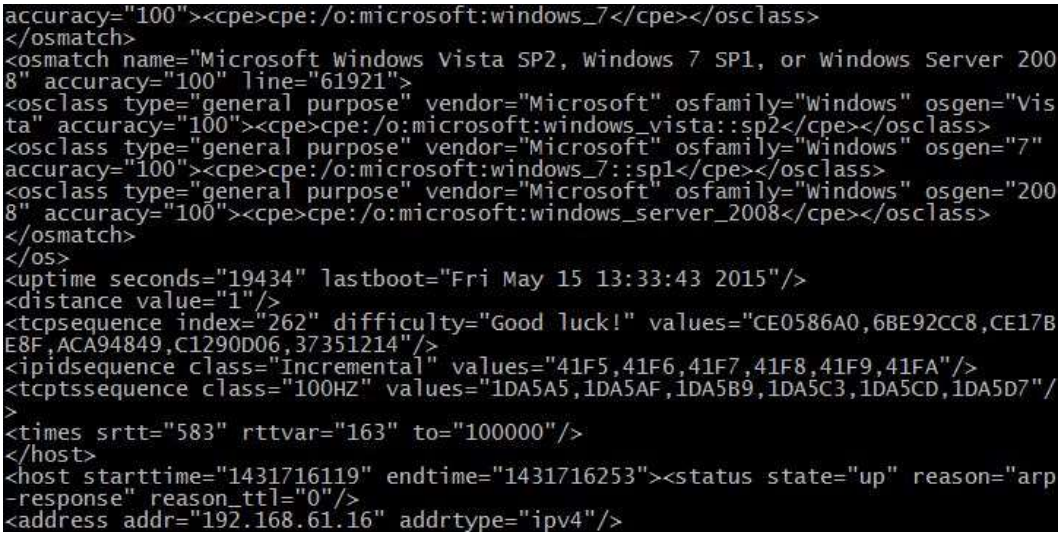


Figura 8.12 Extracto del fichero de respuesta de Nmap, para la detección del sistema operativo

Este módulo cuenta con dos propósitos. El primero de ellos es el de tratar toda esa información e incluirla en la base de datos. Para ello, se utiliza una herramienta

llamada analizador sintáctico o parser [19], que permite crear una estructura de datos a partir de un fichero XML, como el mostrado previamente. Con esta estructura de datos, se coteja por el campo *hostname* para obtener los nombres de las máquinas. En base a este campo, mediante expresiones regulares [20] propiamente creadas para la universidad, se extrae el nombre de las aulas de la escuela. Un ejemplo de resultado obtenido con el *hostname* ai109-03.epsevg.upc.es, sería el aula ai109. Todos los sistemas que no dispongan de un nombre contemplado por las expresiones regulares, se añaden a un grupo genérico llamado *Unknown*. En la Figura 8.13 se pueden observar los grupos que hay actualmente en la aplicación dentro del entorno de la universidad, todos ellos creados automáticamente.

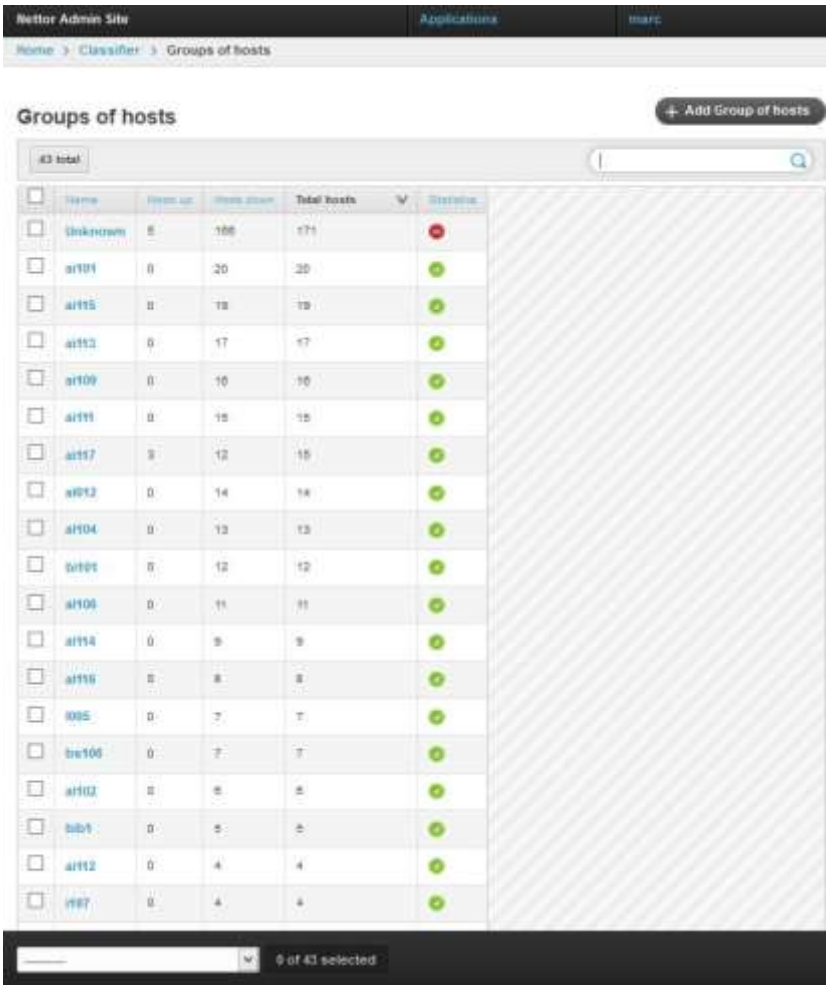


Figura 8.13 Listado de grupos existentes en la aplicación desplegada en la universidad

Una vez disponible toda esta información tratada, se consulta la base de datos y si no existe ningún registro para un equipo, identificándose por su dirección MAC, se añade. En caso de que exista un registro previo y aparezca entre los resultados, se actualiza su información. Para todos los sistemas que no han sido detectados, se les supone como inactivos, no conectados a la red.

Una vez se dispone de la información de los sistemas operativos usados por los sistemas conectados a la red, esta se actualiza en la base de datos. Cabe destacar, que para cada sistema detectado, se mantiene la fecha en la que se ha detectado por última vez. Como se verá en el apartado del módulo de alertas, esta información será crucial para su funcionamiento.

El otro propósito de este módulo, es el de permitir al administrador organizar conjuntos de equipos de forma personalizada, pudiendo crear grupos de sistemas y añadiendo equipos a estos. La Figura 8.14 y la Figura 8.15 muestran esto mismo.

Figura 8.14 Creación manual de un grupo de equipos

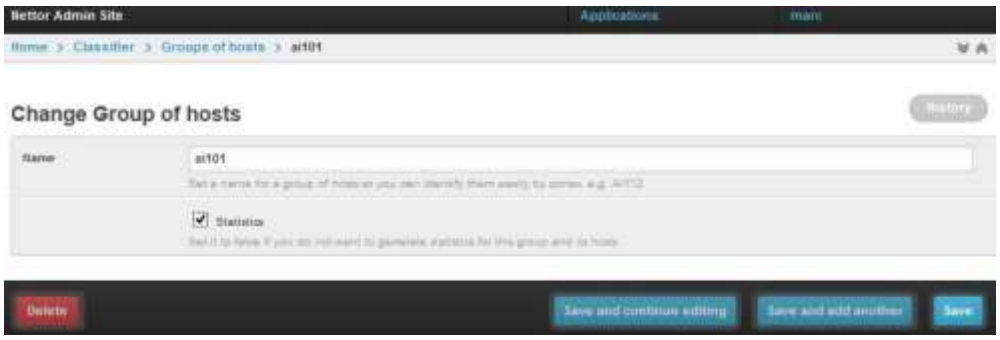


Figura 8.15 Modificación de un grupo de equipos

Esto es útil a la hora de la generación de estadísticas, pues en cada grupo se puede especificar si se quieren obtener estadísticas de los equipos que pertenecen a él o no. Lo mismo ocurre con las alertas, permitiendo seleccionar de qué grupos de dispositivos se desea obtener notificaciones.

Dicho todo esto, también se cuenta con la capacidad de personalizar los nombres de los equipos detectados y como ya se ha mencionado, el grupo al que pertenecen. Para que esta información introducida manualmente no sea sobrescrita por la aplicación en cada escaneado, se ha añadido una opción de bloqueo para que se mantenga. Además, este módulo ofrece la posibilidad de añadir sistemas a una lista negra. Con esto se consigue mantener un registro de los equipos no deseados y a la vez estos no interfieren con la aplicación. Si los equipos no deseados se eliminaran directamente,

en futuros escaneados podrían volver a surgir, pudiendo entorpecer el trabajo de los administradores. Esto último, se puede observar en la Figura 8.15 y la Figura 8.16.

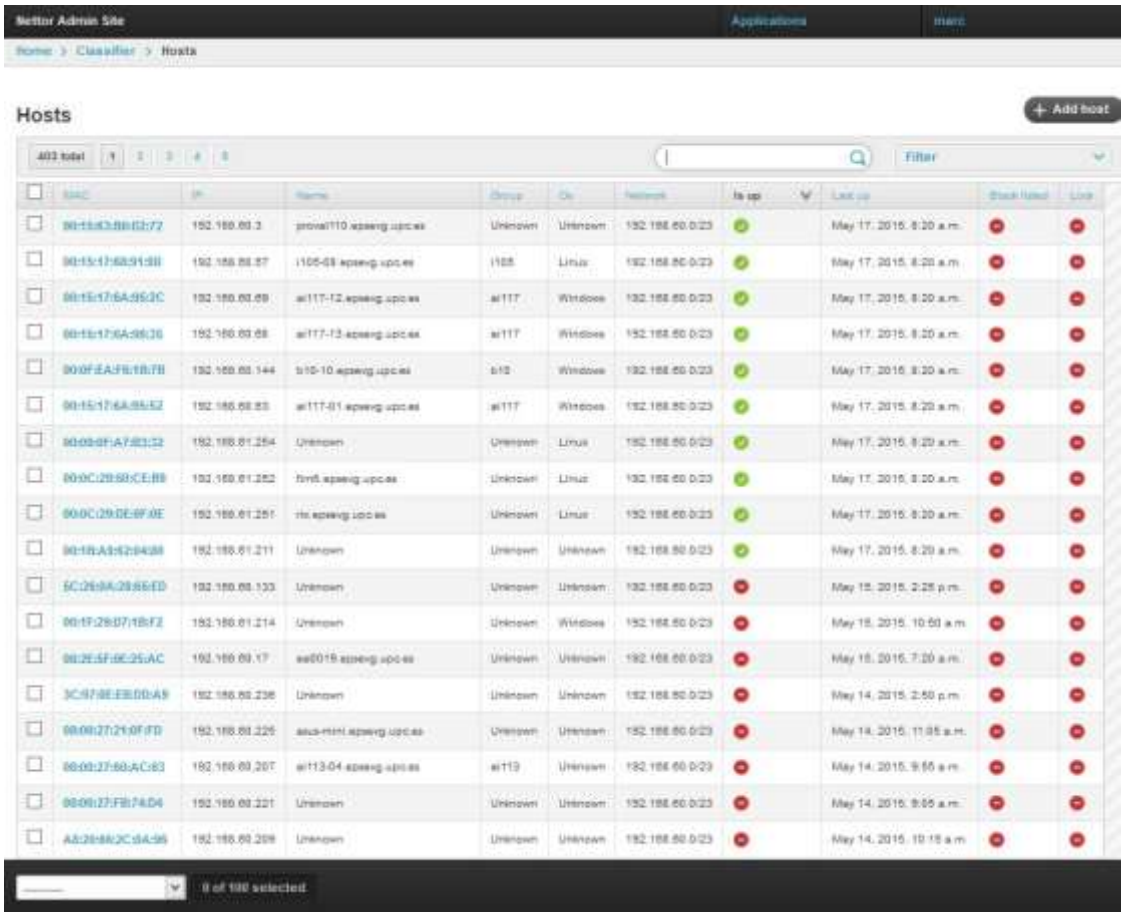


Figura 8.15 Listado de los sistemas detectados

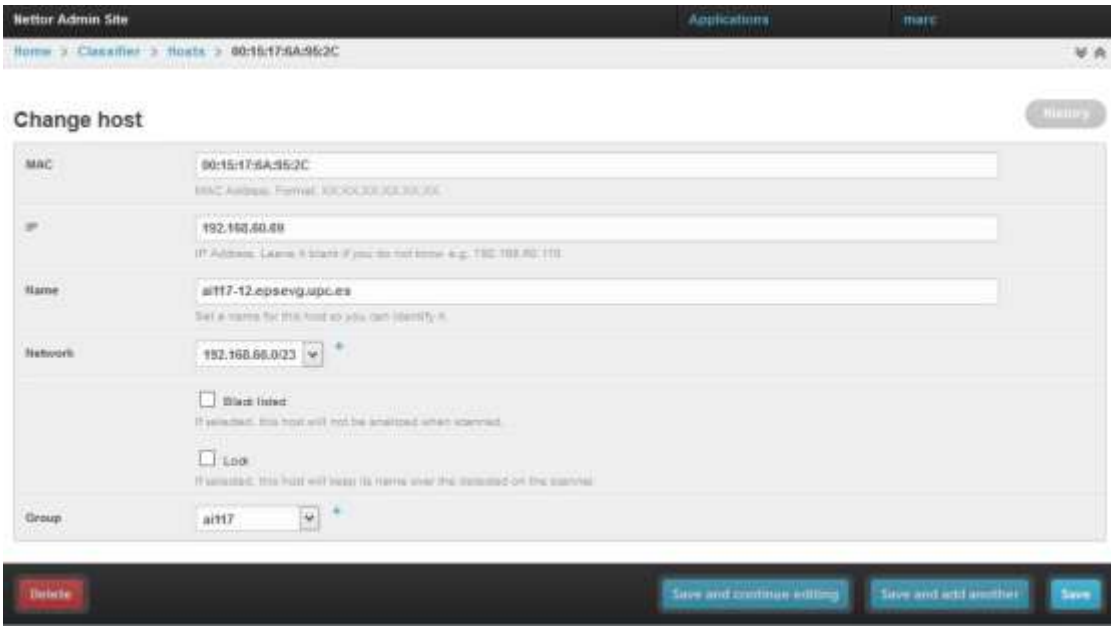


Figura 8.16 Opciones disponibles para la modificación de un equipo

8.1.2.1 Diagrama de flujo

A continuación se muestra el diagrama de flujo que sigue este módulo.

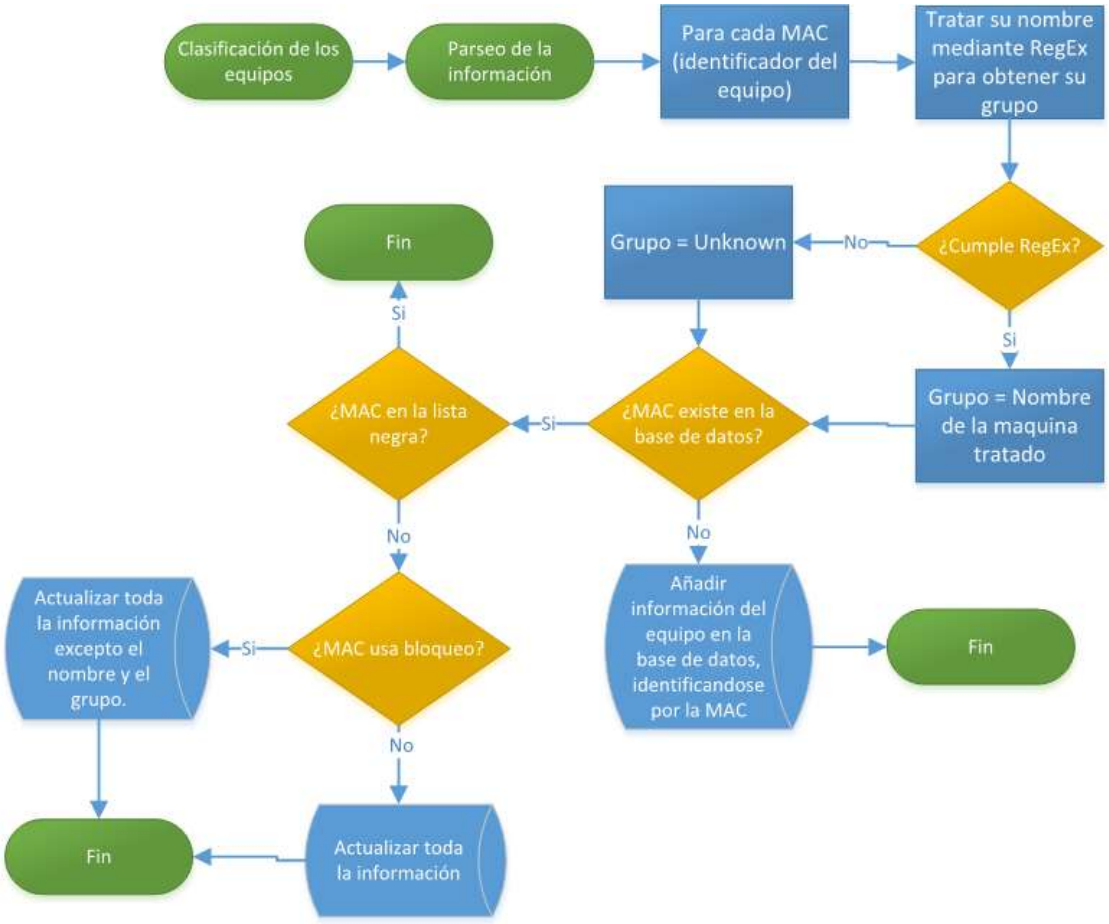


Figura 8.17 Clasificación de los equipos detectados

8.1.3 Módulo de estadísticas

Este módulo se encarga de recopilar la información obtenida de los escaneos, con el fin de generar estadísticas tanto para los equipos, como para los grupos de equipos. Para ello, se hace uso de los escáneres habilitados para recopilar estadísticas. Cuando se ejecuta un escáner de este tipo y finaliza la ejecución del módulo de clasificación, es cuando empieza a trabajar este módulo. Se ofrece la posibilidad de realizar extracciones de datos con el formato CSV, permitiendo al administrador importar los datos a hojas de Excel y poder interactuar con ellos.

Con tal de dar veracidad a las estadísticas generadas, ha sido necesario realizar un estudio sobre el tiempo medio de duración de un escaneo. Se han realizado varios escaneos durante varios días, en distintos momentos, dónde la congestión de la red iba variando. Después de varias pruebas, se ha calculado que el tiempo medio de escaneado es de 3 minutos. Para asegurar su correcta ejecución, se ha establecido un tiempo mínimo entre escaneos de 5 minutos, obteniendo un número de muestras por hora y equipo de 12. Además, se ha establecido que el tiempo máximo entre escaneos sea de 20 minutos, en cuyo caso, el número de muestras por hora y equipo sería de 3. Como se entiende, como más muestras se disponga de un equipo, mayor veracidad tendrán las estadísticas calculadas para este.

En la Figura 8.18 se muestra como está organizado este módulo.

Figura 8.18 Tablas pertenecientes a este módulo

Por un lado contamos con la tabla *Historic_Hosts*. Para todos los equipos que pertenezcan a un grupo con la opción de estadísticas habilitada y que no se encuentren en la lista negra; se crea un registro (una muestra) en este histórico. Aquí, se detalla toda la información de los sistemas (nombre, IP, MAC, sistema operativo, estado y grupo al que pertenece) y el momento exacto en el que se ha añadido. En la Figura 8.19 se puede ver un ejemplo de entradas en el histórico.

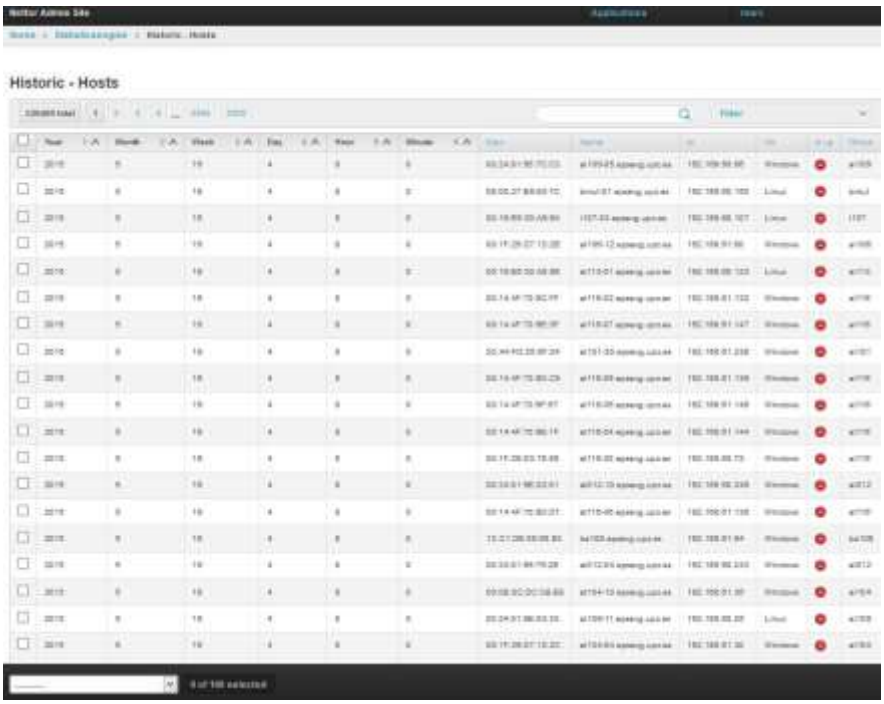


Figura 8.19 Ejemplo de contenido de la tabla de histórico de equipos

En base a la información de este histórico, para cada equipo, se calcula la diferencia de tiempo entre escaneados. Esta información calculada, se almacena en la tabla de estadísticas encargada de mantener un registro de los equipos por horas, *Hosts_Year_Month_Week_Day_Hour*. En esta tabla se indica el número de minutos que han sido detectados en cada una de ellas. En la Figura 8.20 se puede ver un ejemplo de esta tabla. Se puede observar que en la tabla aparecen porcentajes. En el caso de esta, el porcentaje de actividad, se calcula sobre los sesenta minutos que dispone una hora y el número de minutos que ha sido detectado el equipo. El porcentaje de uso del sistema operativo, se calcula sobre el número de minutos en cada uno de ellos sobre el tiempo total de actividad de una máquina.

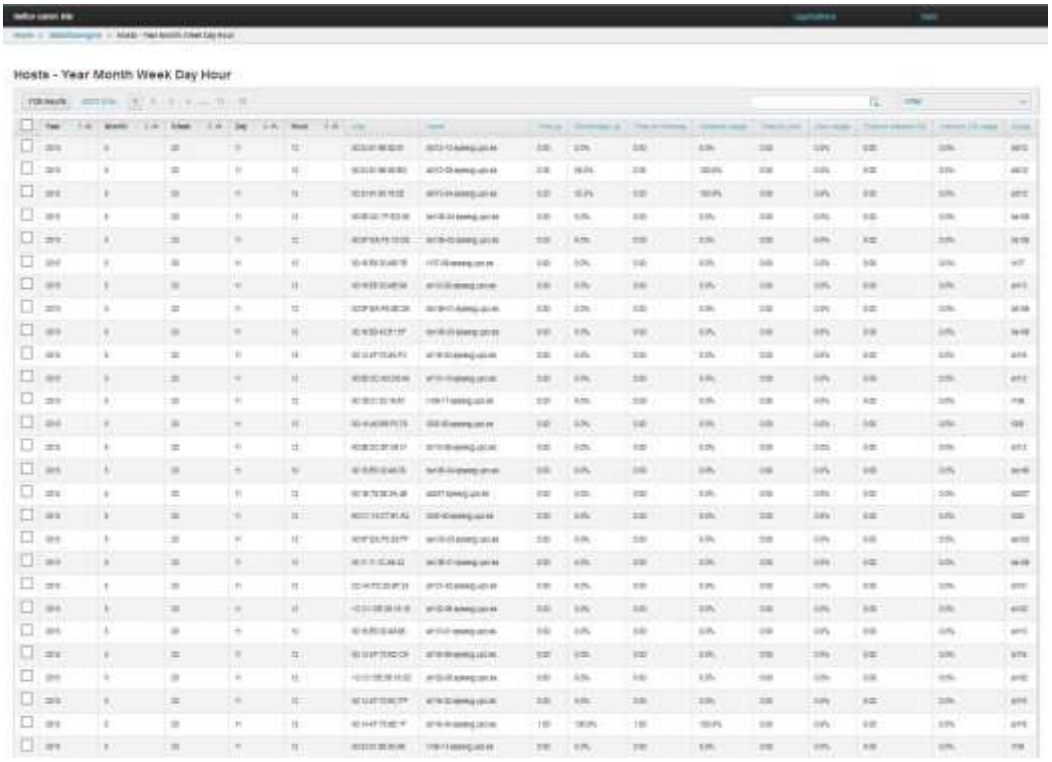


Figura 8.20 Ejemplo de contenido de la tabla de equipos por horas

El siguiente paso es el registro de entradas en la tabla de grupos por hora. Acumulando los valores de los equipos de un mismo grupo y una misma hora, se obtiene el uso total de un grupo durante una hora. En la Figura 8.21 se puede ver un extracto de esta tabla.

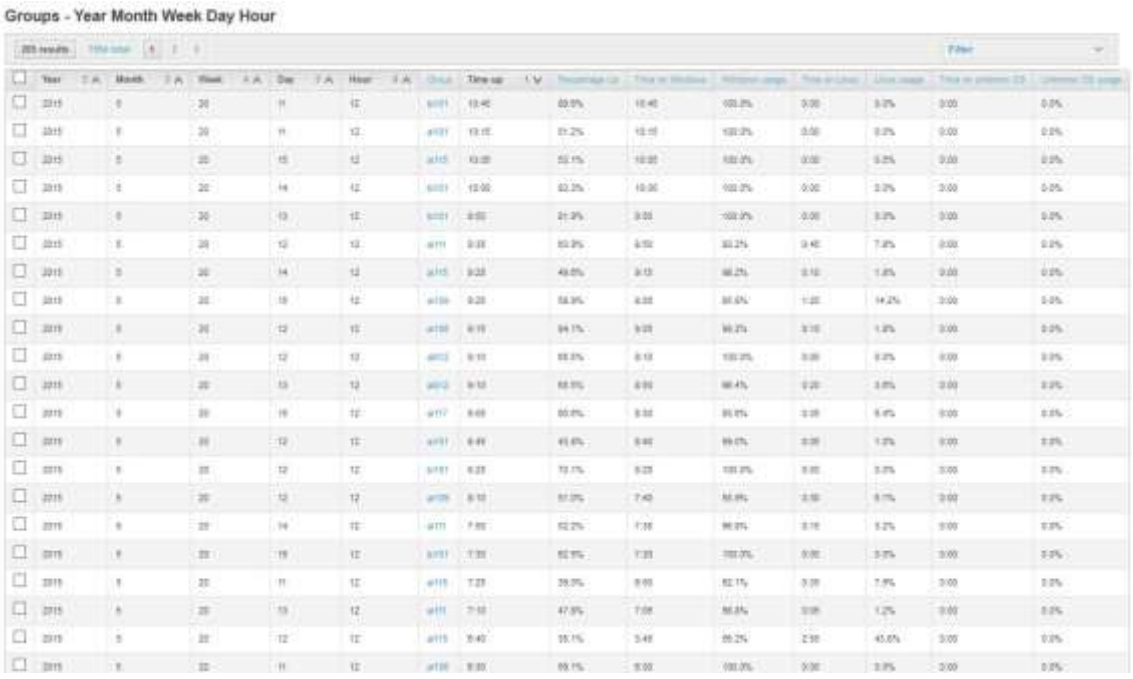


Figura 8.21 Ejemplo de contenido de la tabla de grupos por horas

En la imagen anterior se puede observar cual es la finalidad de todo esto proceso tan complejo de explicar y de entender. Se aprecia que el día 15 de mayo a las 12 horas en el grupo de equipos *bi101*, se estaba haciendo prácticamente uso de todo los equipos disponibles en él. Esto se conoce puesto que porcentaje se calcula en base al número total de sistemas pertenecientes al grupo y el número de minutos que han sido detectados durante ese tiempo.

El resto de tablas son esencialmente datos acumulados de sus predecesoras. Por ejemplo, la tabla de equipos por día, contiene un acumulado de toda la información disponible en la tabla de equipos por hora, tomando como referencia el día en cuestión. Lo mismo ocurre con la tabla de equipos por semana que contiene un acumulado, de toda la información disponible de equipos por día, tomando como referencia la semana en cuestión.

En la Figura 8.22 se muestra la tabla de configuración mencionada al inicio de este apartado. En ella los administradores deben configurar el tiempo que se ejecuta el escáner de estadísticas. Estos valores introducidos se usan en el momento de calcular los porcentajes que se ven en las otras tablas.

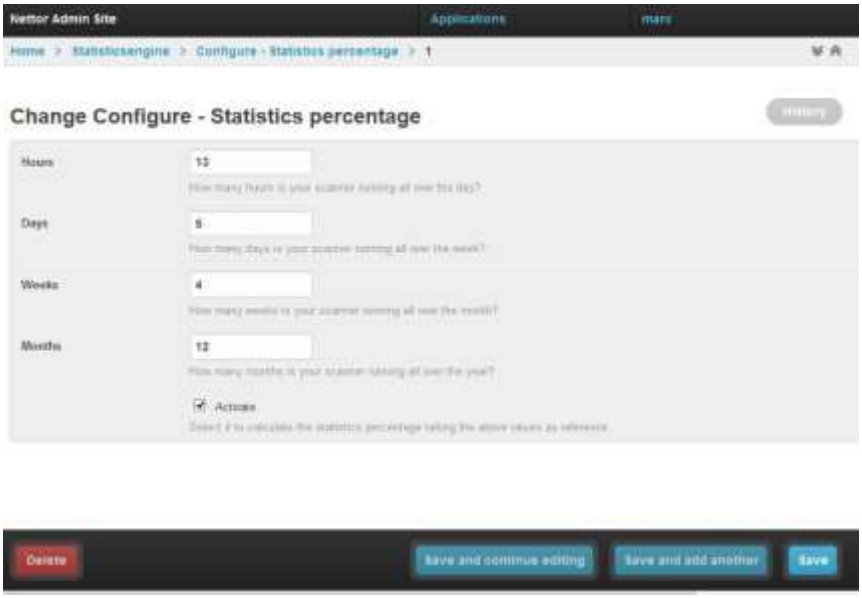
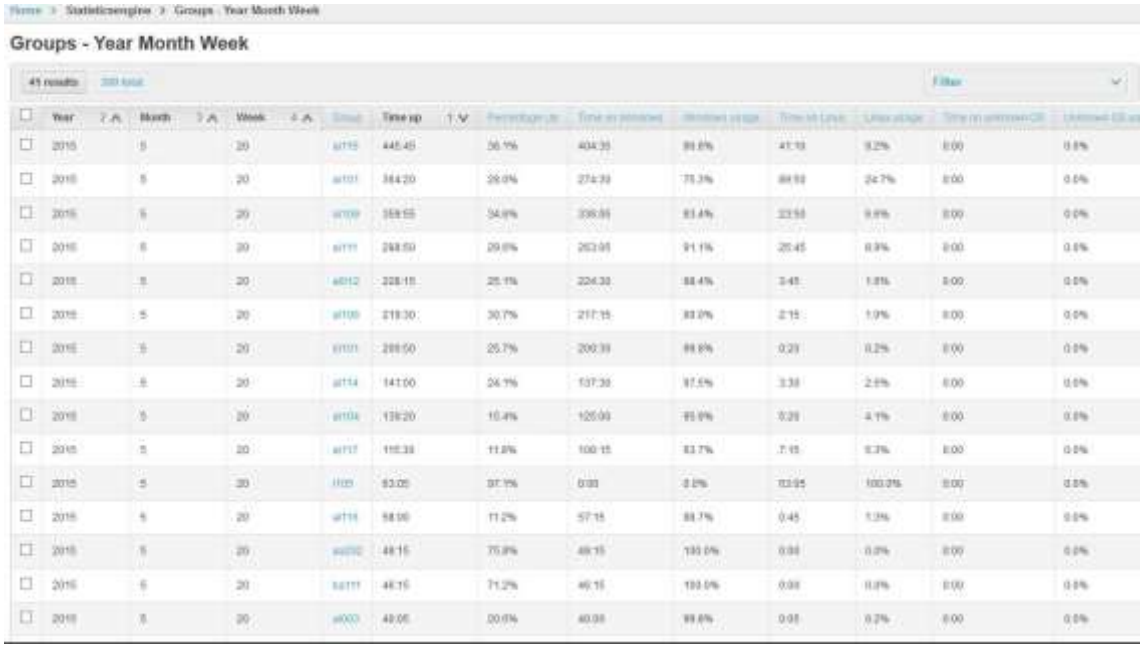


Figura 8.22 Configuración de los porcentajes de las estadísticas

A continuación, con tal de ofrecer un poco de visibilidad real, en la Figura 8.23 se muestra un extracto de la semana 20, indicando el uso que se dio a las aulas informáticas de la escuela durante periodo.



Groups - Year Month Week

Year	Month	Week	Time up	Time on unknown	Time on Linux	Time on unknown OS
2015	5	20	445:45	404:30	41:15	0:00
2015	5	20	364:20	274:30	89:50	0:00
2015	5	20	359:55	336:00	23:55	0:00
2015	5	20	288:50	263:05	25:45	0:00
2015	5	20	228:15	224:30	3:45	0:00
2015	5	20	219:30	217:15	2:15	0:00
2015	5	20	200:50	200:30	0:20	0:00
2015	5	20	141:00	137:30	3:30	0:00
2015	5	20	139:20	125:00	5:20	0:00
2015	5	20	115:30	100:15	7:15	0:00
2015	5	20	93:05	0:00	93:05	0:00
2015	5	20	58:00	57:15	0:45	0:00
2015	5	20	46:15	46:15	0:00	0:00
2015	5	20	46:15	46:15	0:00	0:00
2015	5	20	40:05	40:00	0:05	0:00

Figura 8.23 Extracto de la tabla de uso de las aulas por la semana 20

8.1.4 Módulo de alertas

La finalidad de este último módulo, es la de ofrecer la posibilidad de configurar dos tipos de alertas distintos y enviar notificaciones vía e-mail, a los administradores registrados en la aplicación.

Este módulo, solo se ejecuta con los escáneres definidos sin la opción de estadísticas habilitada. Esto es así para poder reducir la carga de trabajo del servidor y poder separar funcionalidades distintas.

A continuación se detallará el proceso seguido para el desarrollo del módulo.

8.1.4.1 Alerta de equipos inactivos durante x días

El propósito de esta alerta es el de informar a los administradores, de la existencia de equipos no detectados por más o igual a un número de días establecido. Su utilidad reside en poder aportar proactividad a los administradores, notificándoles de posibles averías en algunos equipos de la red. Así, se consigue mejorar el servicio ofrecido, pudiéndose adelantar a las posibles quejas por parte de los usuarios, sobre el mal estado de los equipos de la escuela.

Para desempeñar esta tarea, se han tenido que dividir las funcionalidades. Por un lado se cuenta con la creación y configuración de este tipo de alertas. En este punto, los administradores pueden introducir un nombre para identificar a la alerta, el número de días que debe permanecer un equipo sin detectarse en la red, los grupos de máquinas que se quieren mantener controlados, la hora a la que se quiere recibir la notificación diaria y la opción de activar o desactivar la alerta. En la Figura 8.24 se pueden observan las opciones que se le ofrecen al administrador.

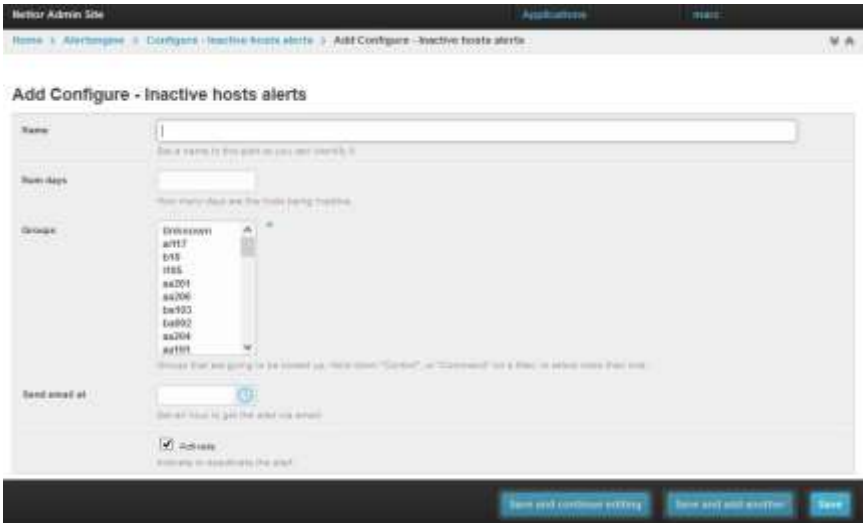


Figura 8.24 Configurando una nueva alerta de equipos inactivos

Como ocurre en los módulos anteriores, se permite consultar las configuraciones creadas, modificarlas e incluso eliminarlas si ya no se requiere de su uso. Esto se puede observar en la Figura 8.25 y la Figura 8.26.

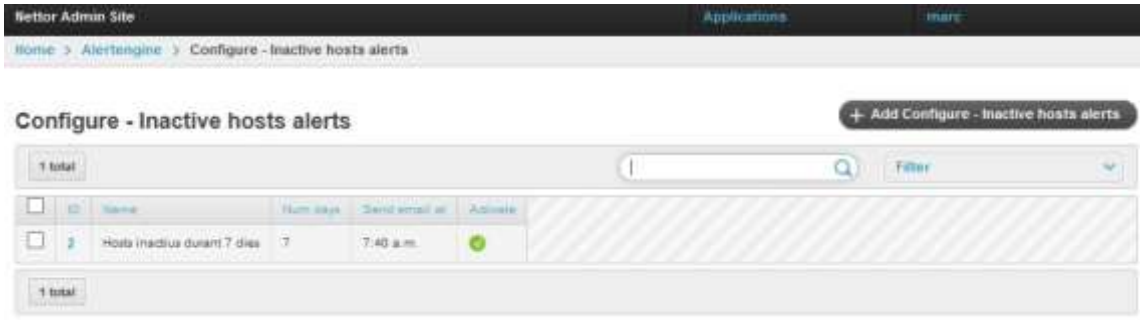


Figura 8.25 Listado de las configuraciones de alertas de equipos inactivos

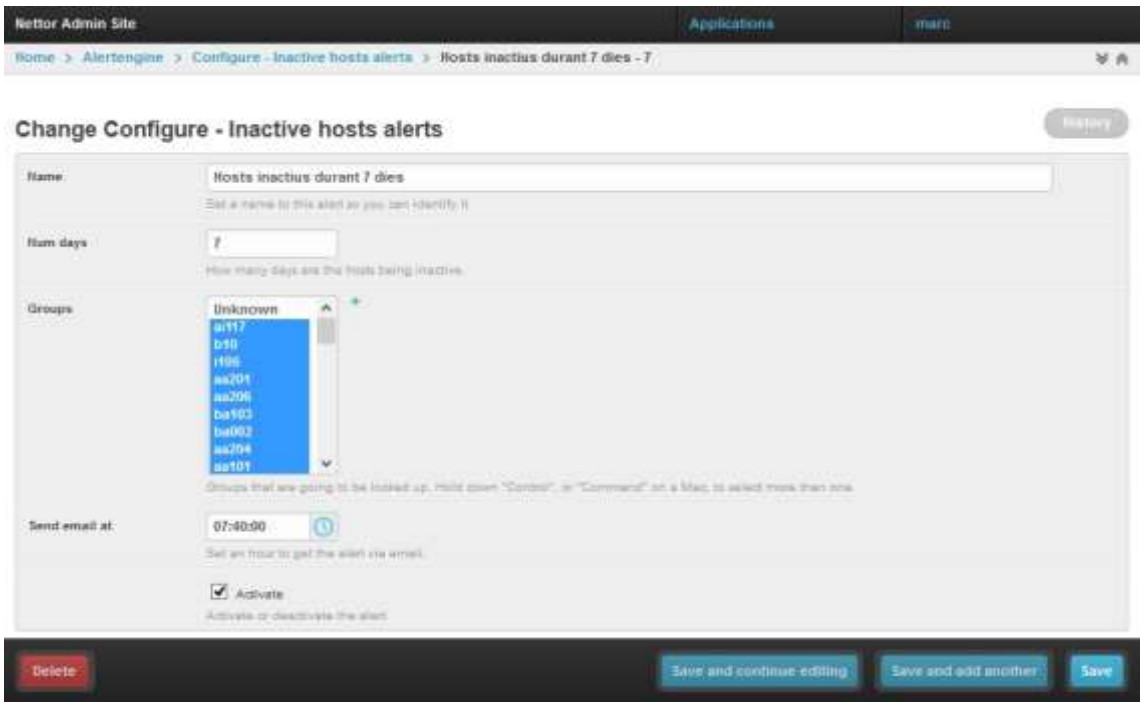


Figura 8.26 Consultando la configuración de una alerta de equipos inactivos, pudiéndose modificar y eliminar

Cuando una alerta se encuentra activada, al final de la ejecución de los escáneres, se comprueba la fecha de ultima detección de los equipos; pertenecientes a los grupos seleccionados en la configuración. Si esta fecha es anterior al número de días especificados en la configuración, en base de la fecha de ejecución del escáner ejecutado, se añade un registro en del sistema. Con tal de mantener un registro de todas las detecciones que se han realizado, se ha creado una tabla dentro de la base de datos que contiene la información relacionada para cada sistema y la alerta activada.

Con el fin de complementar la explicación anterior, en la Figura 8.27 se pueden observar una serie de equipos de la red de aulas informáticas de la EPSEVG, los cuales no han sido detectados por la aplicación en 7 días o más.

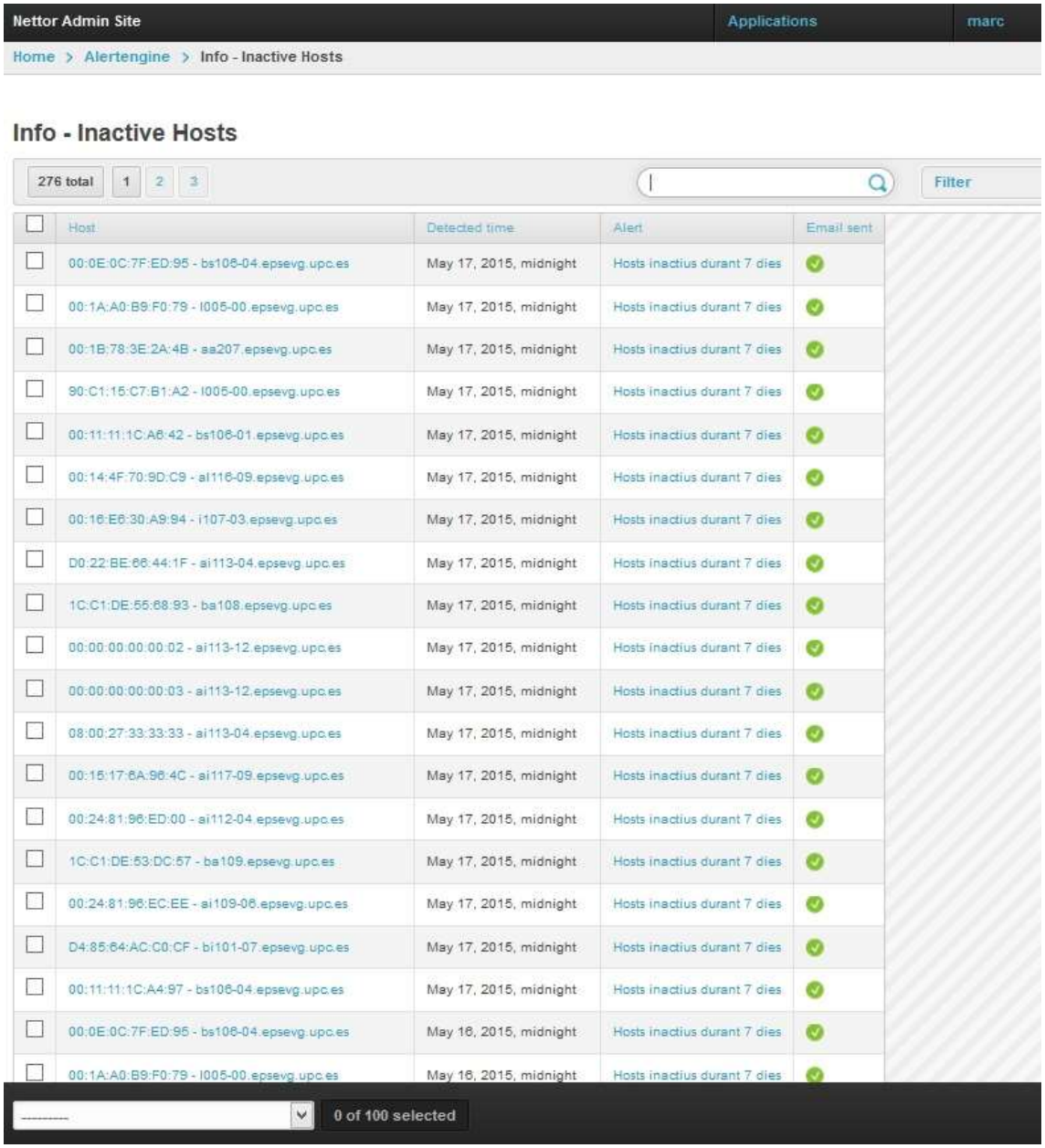


Figura 8.27 Registro de los equipos que han disparado la alerta

En la Figura 8.28 se muestra un ejemplo de correo de notificación para la alerta anterior.

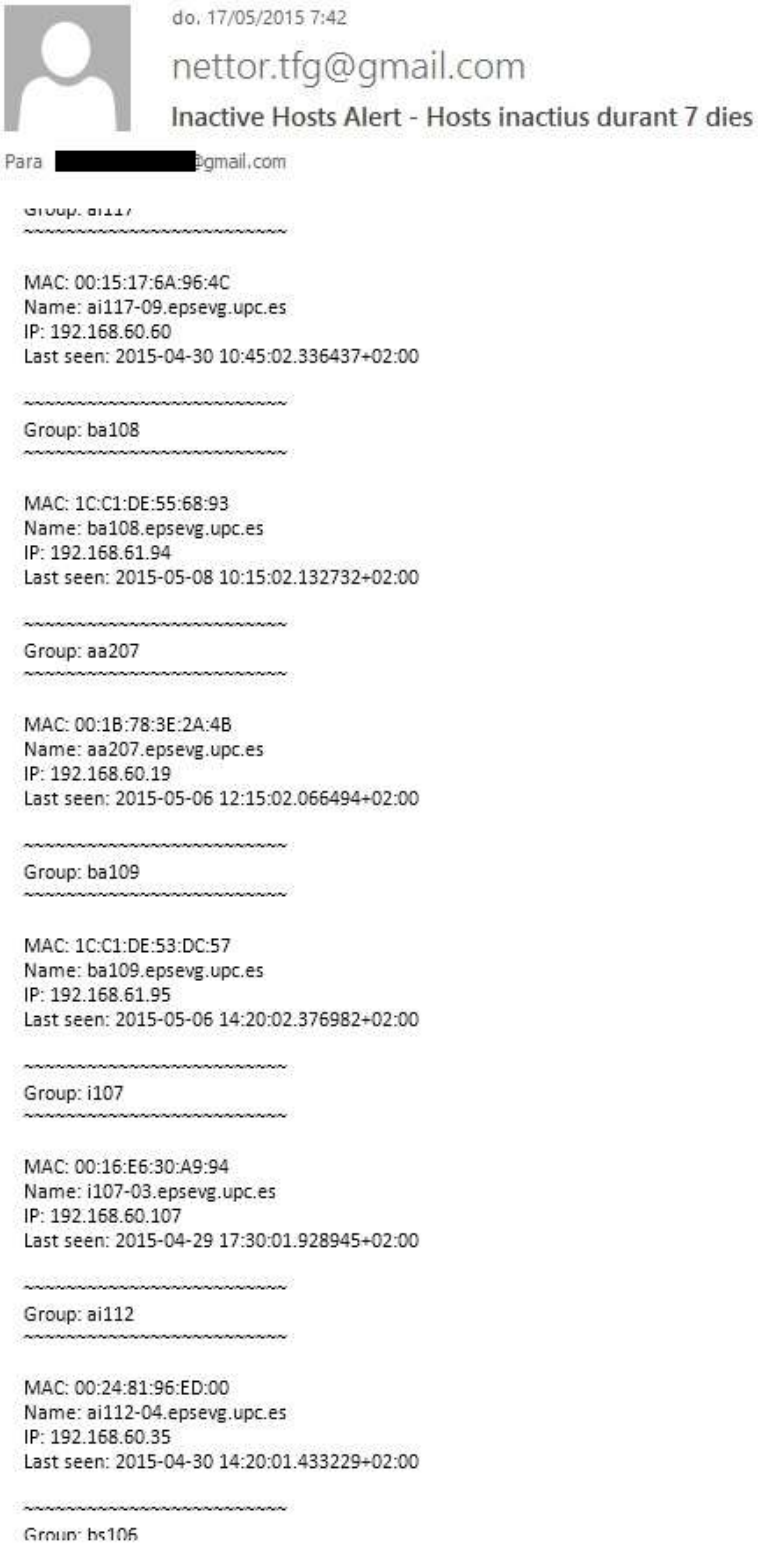


Figura 8.28 Ejemplo de correo de notificación de equipos inactivos durante 7 días o más

8.1.4.2 Alerta de equipos activos entre rango de horas

El propósito de esta alerta es el de informar a los administradores, de la existencia de equipos conectados a la red, dentro de un rango de horas establecido. Su utilidad reside en detectar equipos en horas en las que no debería encontrarse ninguna máquina encendida en la escuela. Además, con una buena organización de los equipos clasificados en grupos, se podría aprovechar para la detección de equipos ajenos a la escuela en horas fuera de lo habitual. Este último caso, podría significar la presencia de alguna maquina potencialmente peligrosa dentro del entorno de la escuela.

Su creación es muy similar al otro tipo de alerta mostrada. Cada alerta creada dispone de un nombre que la identificará, el rango de horas a querer revisar, los grupos que se quieren tener en cuenta, en que momento del día se desea obtener la notificación y si se quiere activar o no. Es importante mencionar, que en este tipo de alertas, la hora de envío de la notificación, debe estar dentro del rango de horas revisadas. En la Figura 8.29 se muestra un ejemplo de alerta configurada.

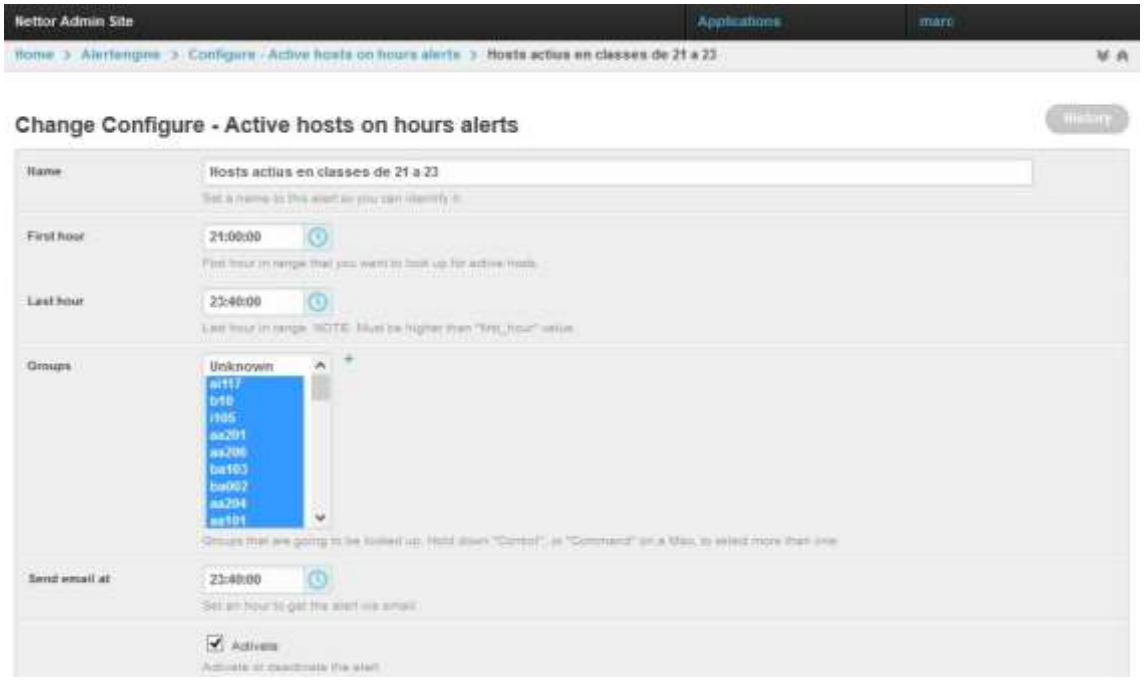


Figura 8.29 Ejemplo de configuración de una alerta de hosts detectados entre un rango de horas

El funcionamiento interno también es muy similar al anterior tipo de alerta. Cuando un escáner se está ejecutando dentro del rango de horas establecido por una alerta, los equipos detectados se registran en una tabla para tener constancia de ellos.

En la Figura 8.30 se puede observar un ejemplo de esta tabla. En ella se observan dos tipos de alertas registradas con varios sistemas detectados.

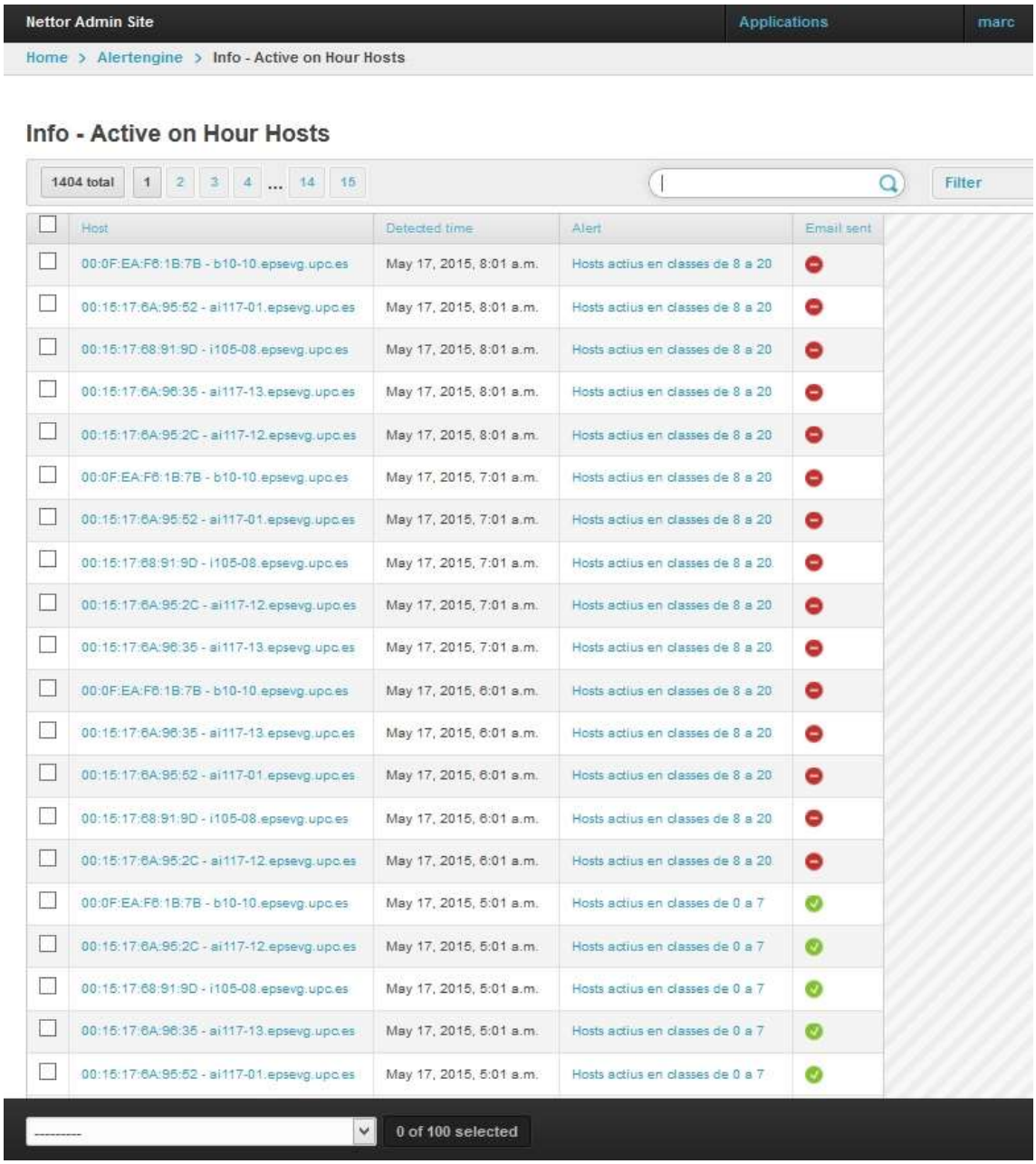


Figura 8.30 Información sobre los equipos detectados por dos alertas distintas

En la Figura 8.31 se muestra un ejemplo de correo de notificación para este tipo de alertas.

do. 17/05/2015 7:42

nettor.tfg@gmail.com

Active Hosts on hours alert - Hosts actius en classes de 0 a 7

Para [redacted]@gmail.com

.....
Hour: 4
.....

MAC: 00:15:17:68:91:9D
Group: i105
Name: i105-08.epsevg.upc.es
IP: 192.168.60.57

MAC: 00:15:17:6A:95:52
Group: ai117
Name: ai117-01.epsevg.upc.es
IP: 192.168.60.63

MAC: 00:0F:EA:F6:1B:7B
Group: b10
Name: b10-10.epsevg.upc.es
IP: 192.168.60.144
.....
Hour: 5
.....

MAC: 00:15:17:68:91:9D
Group: i105
Name: i105-08.epsevg.upc.es
IP: 192.168.60.57

MAC: 00:15:17:6A:95:52
Group: ai117
Name: ai117-01.epsevg.upc.es
IP: 192.168.60.63

MAC: 00:0F:EA:F6:1B:7B
Group: b10
Name: b10-10.epsevg.upc.es
IP: 192.168.60.144

MAC: 00:15:17:6A:95:2C
Group: ai117
Name: ai117-12.epsevg.upc.es
IP: 192.168.60.69

MAC: 00:15:17:6A:96:35

Figura 8.31 Ejemplo de correo de notificación de activos entre las 0 y las 7

8.1.4.3 Configuración del servidor de correos

Para que la aplicación sea capaz de enviar emails, es necesario especificarle que servidor y cuenta de correo se debe utilizar. Para ello, se ha añadido a la aplicación la capacidad de configurar un servidor de correo. En la Figura 8.32, se puede ver la configuración establecida para el servidor de correo de pruebas.

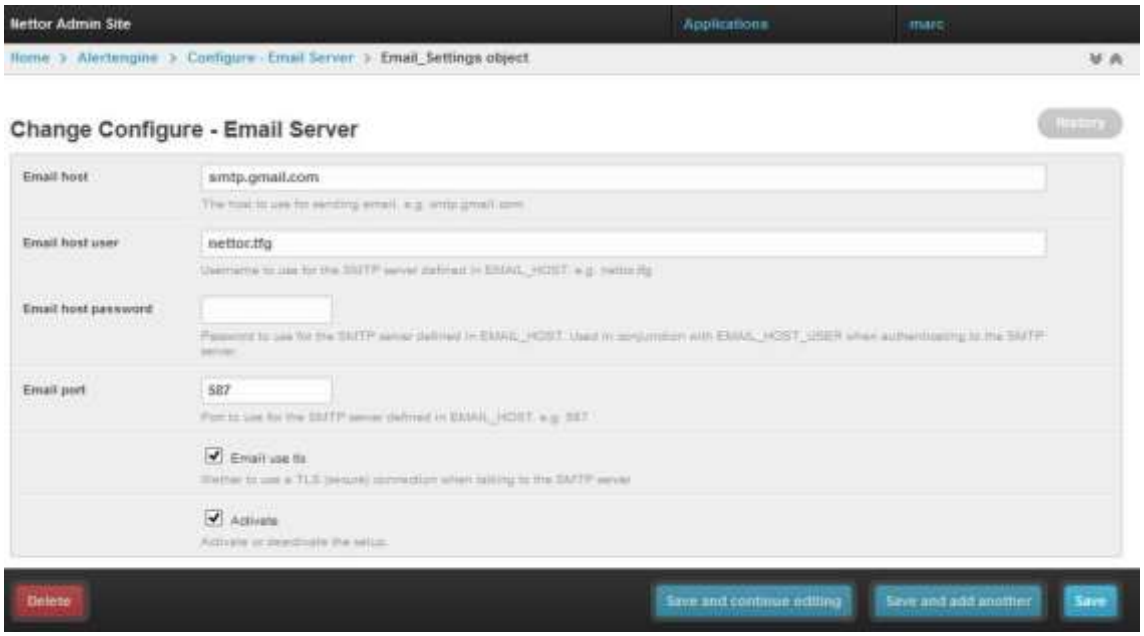


Figura 8.32 Configuración usada para el envío de correos

8.2 Módulo de autenticación

Django incorpora en su propio motor, un sistema de autenticación de usuarios. Para este proyecto se ha integrado este sistema. Con esto se consigue por un lado, lo mostrado en el capítulo anterior, ofreciendo una pantalla de login para poder acceder a la aplicación.

Este sistema, también es usado en la notificación de alertas por correo, pues sus destinatarios son los usuarios registrados en él. Por esto es importante que durante el registro de todos los administradores, se especifique una dirección de correo válida para poder recibir las alertas deseadas.

8.3 Diagrama de clases

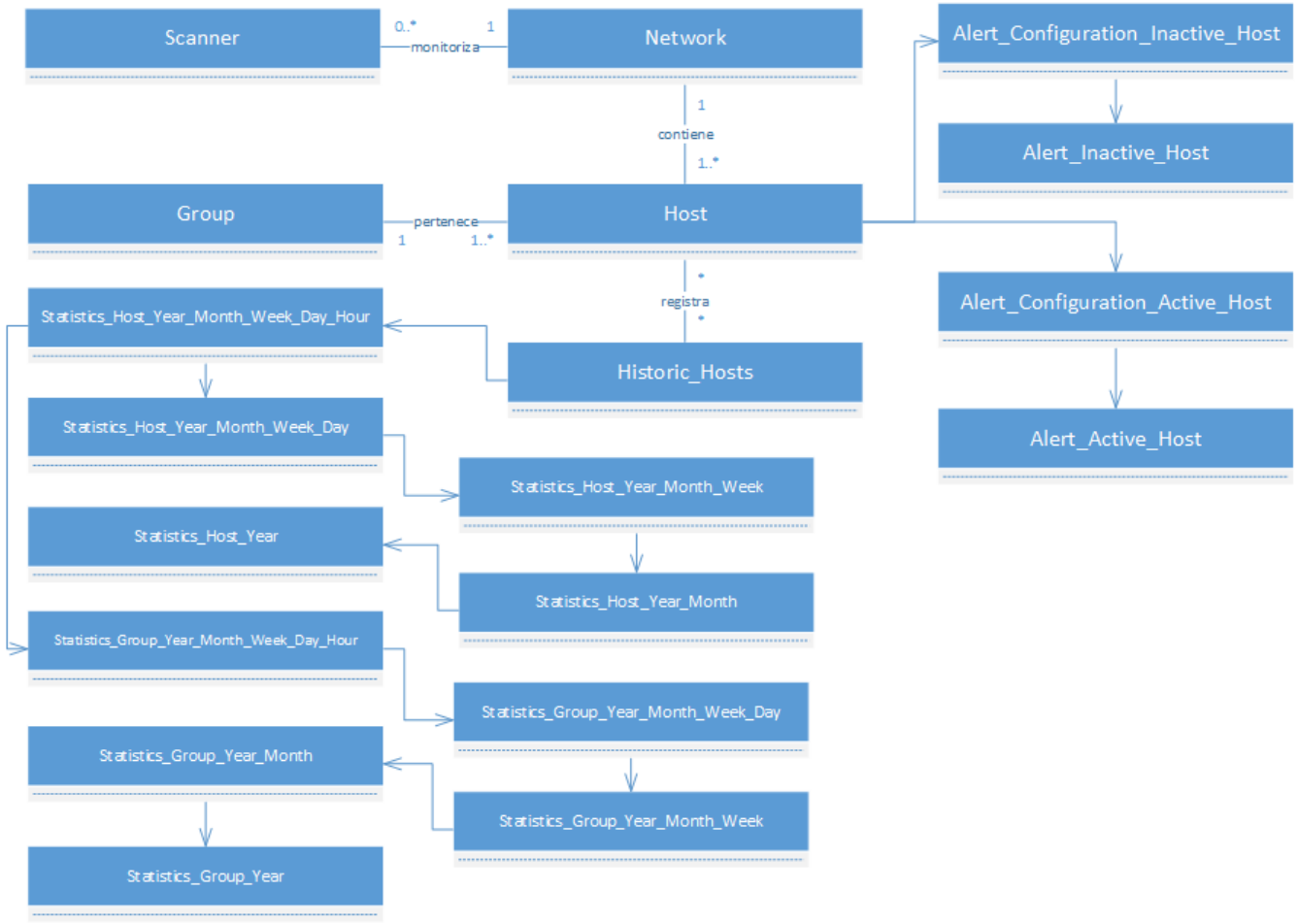


Figura 8.33 Diagrama que muestra las relaciones entre las principales clases desarrolladas a lo largo del proyecto

9. INTEGRACIÓN

En este capítulo se documenta el proceso de integración de la herramienta desarrollada, en el servidor dedicado. Se documenta como se han configurado los distintos componentes vistos en el capítulo 6, para funcionar en harmonía con la aplicación. Todos los tecnicismos, tales como las instrucciones realizadas y las configuraciones, se encuentran en los anexos.

9.1 Pasos previos

Antes de empezar a configurar los diversos componentes, se deben efectuar una serie de acciones previas en el servidor.

- Creación del usuario *Nettor*, propietario de la aplicación web y de la base de datos. Se debe crear un usuario con los mínimos privilegios posibles en el sistema, por temas de seguridad.
- Revisión de los servidores DNS configurados. Si no están bien configurados, la aplicación no será capaz de realizar las consultas inversas a los DNS, para poder detectar los nombres de las máquinas.
- Revisión de las interfaces de red. Se debe comprobar que el servidor forme parte de las redes que se quieran monitorizar, estando estas presentes en el servidor en forma de interfaz.

9.2 Configuraciones

9.2.1 Lenguaje de programación - Python

Se debe instalar la versión 3.4.2 de Python en el servidor. Una vez instalada, se debe crear un entorno virtual donde residirá la aplicación web.

9.2.2 SGBD - PostgreSQL

La configuración de PostgreSQL es muy sencilla. Simplemente se debe crear una base de datos, nombrada *NettorDB*, la cual será la contenedora de toda la información de la aplicación. El usuario propietario de esta base de datos, debe ser *Nettor*.

9.2.3 Herramienta de detección - Nmap

Para poder utilizar la detección de equipos mediante consultas ARP, se requieren privilegios de root. Para ello, se deberá crear una entrada en el fichero de *sudoers*, especificando que el usuario *Nettor* dispone de permisos totales para ejecutar Nmap, sin solicitud de contraseña.

9.2.4 Servidor web – Nginx

Esta parte es la más delicada de configurar. Por un lado se configura el servidor para que trabaje únicamente con conexiones cifradas, usando HTTPS. Esto permitirá cifrar la conexión entre el cliente y el servidor, protegiendo la integridad y la confidencialidad de los datos transmitidos. Simplemente con la instalación de *mod_ssl* y la creación de unos certificados, se puede habilitar esta opción.

Por el otro lado se debe configurar la comunicación con la aplicación web mediante la interfaz gunicorn.

9.2.5 Aplicación web - Nettor

La aplicación web se encuentra comprimida en un fichero tar.gz en los anexos. Para integrarla en un nuevo servidor, simplemente se debe extraer su contenido en la carpeta /var/www/nettor/. Una vez estén todos los ficheros en esa carpeta, se deben cambiar los permisos del directorio y subdirectorios además del propietario de los mimos.

Se hace uso de una herramienta nombrada *supervisor* para poder levantar el servidor en caso de desconexiones causadas por cortes de luz o de línea.

El fichero /var/www/nettor/nettor/settings.py, contiene el corazón de la aplicación. Ahí se especifica valores clave para el funcionamiento de la herramienta tales como la conexión con la base de datos.

9.3 Distribución del sistema

Es posible distribuir el sistema de dos formas distintas.

- **Máquina virtual.** Ya que todo el desarrollo se ha realizado en una máquina virtual, se puede hacer una copia de esta y reutilizarla. En caso de seleccionar esta opción, únicamente se tendría que tener en cuenta los dos últimos pasos descritos en el apartado de pasos previos.
- **Instalación manual.** En los anexos se detallan las instrucciones llevadas a cabo para la instalación de todos los componentes y sus ficheros de configuración. Así como de la aplicación desarrollada. Simplemente se tendrían que tener en cuenta los dos apartados anteriores y la información disponible en los anexos para su correcta instalación.

10 Resultados

Se ha desarrollado una aplicación web segura, que junto a los componentes analizados, forma un sistema de monitorización que cumple las expectativas. El sistema obtenido permite la visualización en tiempo real de los equipos monitorizados, a través de una interfaz minimalista. Además es capaz de generar estadísticas de uso de los equipos y grupos de equipos, permitiendo a los administradores analizar la información recopilada. También, cuenta con la posibilidad de generar alertas, notificando a los administradores vía correo y acceder a todos los datos y configuraciones mediante una interfaz de administración. Finalmente se ofrecen elementos personalizables tales como la creación de grupos de equipos y la configuración de sistemas de escaneado.

En definitiva, el proyecto cumple con todos los objetivos fijados, puesto que tras su despliegue en el servidor de la escuela ha funcionado tal y como estaba planeado.

11 Trabajo futuro

Existen posibles mejoras para este trabajo, a continuación se indican algunas de ellas:

- Incorporación de Ajax para poder visualizar la información justo en el momento en que aparezca en la base de datos. Así se podría eliminar la etiqueta temporizadora de HTML presente en la interfaz básica.
- Generación de graficas en base a los datos de las estadísticas. Los datos recopilados y tratados se podrían ofrecer de una forma más visual dentro de la aplicación, sin tener que depender de herramientas externas como Excel.
- Nuevos tipos de alertas. Por ejemplo se podría implementar un tipo de escaneado que realizase detecciones de servicios en las máquinas. En caso de detectar algún servicio fuera de los parámetros establecidos, se generaría una alerta.
- Interacción directa con el servidor. Se podría habilitar una vista en la aplicación web, que permitiera conectarse al servidor mediante SSH.

12 Conclusiones

Con la realización de este proyecto, se han aprovechado los conocimientos adquiridos a lo largo de la carrera. El conocimiento de redes y de la administración de sistemas operativos y sus servicios, han jugado un papel muy importante. Por otro lado, al inicio de este se desconocía completamente, el funcionamiento de las aplicaciones web y de la mayoría de los componentes que forman parte del sistema generado. Esto ha llevado emplear muchísimo tiempo al estudio de todos estos y creo que el resultado final ha valido la pena.

Con este proyecto espero aportar mi granito de arena para mejorar los servicios ofrecidos en la escuela.

13 BIBLIOGRAFÍA

[1] Colaboradores de Wikipedia. Classless Inter-domain Routing [en línea]. Wikipedia, La enciclopedia libre, 2015 [fecha de consulta: 16 de Febrero del 2015]. Disponible en <https://es.wikipedia.org/wiki/Classless_Inter-Domain_Routing>.

[2] Colaboradores de Wikipedia. Address Resolution Protocol [en línea]. Wikipedia, La enciclopedia libre, 2015 [fecha de consulta: 16 de Febrero del 2015]. Disponible en <https://es.wikipedia.org/wiki/Address_Resolution_Protocol>.

[3] Equipo de Nagios. About Nagios [en línea]. Nagios, Página oficial, 2015 [fecha de consulta: 20 de Abril del 2015]. Disponible en <https://www.nagios.org/about>.

[4] Equipo de Icinga. Icinga [en línea]. Icinga, Página oficial, 2015 [fecha de consulta: 20 de Abril del 2015]. Disponible en <https://www.icinga.org/>.

[5] Equipo de Nmap. Nmap description [en línea]. Nmap, Página oficial, 2015 [fecha de consulta: 24 de Febrero del 2015]. Disponible en <https://nmap.org/book/man.html#man-description>.

[6] Colaboradores de Wikipedia. Modelo-Vista-Controlador [en línea]. Wikipedia, La enciclopedia libre, 2015 [fecha de consulta: 26 de Febrero del 2015]. Disponible en <https://es.wikipedia.org/wiki/Modelo%E2%80%93vista%E2%80%93controlador>.

[7] Equipo de Debian. Acerca de Debian [en línea]. Debian, Página oficial, 2015 [fecha de consulta: 26 de Febrero del 2015]. Disponible en <https://www.debian.org/intro/about#what>.

[8] Equipo de CentOS. About CentOS [en línea]. CentOS, Página oficial, 2015 [fecha de consulta: 26 de Febrero del 2015]. Disponible en <https://www.centos.org/about/#centos-linux>.

[9] Equipo de TIOBE. TIOBE Programming Community Index [en línea]. TIOBE, Página oficial, 2015 [fecha de consulta: 27 de Febrero del 2015]. Disponible en <http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html>.

[10] RENEE. Code Wars: Ruby vs Python vs PHP [Infographic] [en línea]. Udemy blog, 2012 [fecha de consulta: 27 de Febrero del 2015]. Disponible en <https://blog.udemy.com/modern-language-wars/>.

[11] IwanVosloo. Web Frameworks for Python [en línea]. Python, Página oficial, 2015 [fecha de consulta: 28 de Febrero del 2015]. Disponible en <https://wiki.python.org/moin/WebFrameworks>.

[12] Equipo de Django. Meet Django [en línea]. Django, Página oficial, 2015 [fecha de consulta: 28 de Febrero del 2015]. Disponible en <https://www.djangoproject.com/>.

[13] Equipo de PostgreSQL. About PostgreSQL [en línea]. PostgreSQL, Página oficial, 2015 [fecha de consulta: 2 de Marzo del 2015]. Disponible en <http://www.postgresql.org/about/>.

[14] Equipo de Django. Django documentation [en línea]. Django, Página oficial, 2015 [fecha de consulta: a lo largo del proyecto]. Disponible en <https://docs.djangoproject.com/en/1.7/>.

[15] Equipo de Nginx. Nginx [en línea]. Nginx, Página oficial, 2015 [fecha de consulta: 2 de Marzo del 2015]. Disponible en <http://nginx.org/en/>.

[16] Colaboradores de Wikipedia. Web Server Gateway Interface [en línea]. Wikipedia, La enciclopedia libre, 2015 [fecha de consulta: 8 de Marzo del 2015]. Disponible en <https://en.wikipedia.org/wiki/Web_Server_Gateway_Interface>.

[17] Equipo de Gunicorn. Gunicorn [en línea]. Gunicorn, Página Oficial, 2015 [fecha de consulta: 8 de Marzo del 2015]. Disponible en <http://gunicorn.org/>.

[18] Colaboradores de Wikipedia. Cron [en línea]. Wikipedia, La enciclopedia libre, 2015 [fecha de consulta: 16 de Marzo del 2015]. Disponible en <https://en.wikipedia.org/wiki/Cron>.

[19] Colaboradores de Wikipedia. Parser [en línea]. Wikipedia, La enciclopedia libre, 2015 [fecha de consulta: 20 de Marzo del 2015]. Disponible en <https://es.wikipedia.org/wiki/Analizador_sint%C3%A1ctico>.

[19] Colaboradores de Wikipedia. Regular expression [en línea]. Wikipedia, La enciclopedia libre, 2015 [fecha de consulta: 23 de Marzo del 2015]. Disponible en <https://en.wikipedia.org/wiki/Regular_expression>.

[20] Equipo de Bootstrap. Bootstrap [en línea]. Bootstrap, Página oficial, 2015 [fecha de consulta: 30 de Marzo del 2015]. Disponible en <http://getbootstrap.com/>.